GEORGIA INSTITUTE OF TECHNOLOGY — OFFICE OF CONTRACT ADMINISTRATION

PROJECT ADMINISTRATION DATA SHEET

(R5895-0A0) [x] ORIGINAL [ ] REVISION NO. _____

Project No. G-36-617 GTRI/~~GKK~~ DATE 2 / 22 / 85

Project Director: Dr. J. L. Kolodner School/~~XXX~~ ICS

Sponsor: U. S. Army Research Office
Research Triangle Park, NC

Type Agreement: SFRC DAAG29-85-K-0023

Award Period: From 12/1/85 To 11/30/87 (Performance) 1/30/88 (Reports)

Sponsor Amount:	This Change	Total to Date
Estimated:	$ 214,753	$ 214,753
Funded:	$ 64,248	$ 64,248

Cost Sharing Amount: $ None Cost Sharing No: N/A

Title: The Role of Experience in Common Sense and Expert Problem Solving

ADMINISTRATIVE DATA

OCA Contact William F. Brown x-4820

1) Sponsor Technical Contact:

Dr. Jagdish Chandra
U. S. Army Research Office
P. O. Box 12211
Research Triangle Park, NC 27709

2) Sponsor Admin/Contractual Matters:

Mr. T. A. Bryant
ONR RR
Georgia Tech

Defense Priority Rating: None shown Military Security Classification: Unclassified
(or) Company/Industrial Proprietary: N/A

RESTRICTIONS

See Attached Gov't. Supplemental Information Sheet for Additional Requirements.

Travel: Foreign travel must have prior approval – Contact OCA in each case. Domestic travel requires sponsor approval where total will exceed greater of $500 or 125% of approved proposal budget category.

Equipment: Title vests with GIT; however, prior Government approval would be required since no equipment was included in the proposal budget.

COMMENTS:

Sponsor I.D. No. 02.102.001.85.008

COPIES TO:

Project Director
Research Administrative Network
Research Property Management
Accounting
Procurement/EES Supply Services
Research Security Services
Reports Coordinator (OCA)
Research Communications (2)
GTRI
Library
Project File
Other A. Jones

FORM OCA 4:383

GEORGIA INSTITUTE OF TECHNOLOGY OFFICE OF CONTRACT ADMINISTRATION

SPONSORED PROJECT TERMINATION/CLOSEOUT SHEET

Date 4/6/88

Project No. G-36-617 School/~~Lab~~ ICS

Includes Subproject No.(s) N/A

Project Director(s) Dr. J. L. Kolodner GTRC/~~GTX~~

Sponsor U. S. Army Research Office/Research Triangle Park, NC

Title The Role of Experience in Common Sense and Expert Problem Solving

Effective Completion Date: 11/30/87 (Performance) 1/30/87 (Reports)

Grant/Contract Closeout Actions Remaining:

- [ ] None
- [x] Final Invoice or Copy of Last Invoice Serving as Final
- [x] Release and Assignment
- [x] Final Report of Inventions and/or Subcontract:
  Patent and Subcontract Questionnaire sent to Project Director [x]
- [x] Govt. Property Inventory & Related Certificate
- [ ] Classified Material Certificate
- [ ] Other _____

Continues Project No. G-36-617 Continued by Project No. _____

COPIES TO:

Project Director
Research Administrative Network
Research Property Management
Accounting
Procurement/GTRI Supply Services
Research Security Services
Reports Coordinator (OCA)
Program Administration Division
Contract Support Division
Facilities Management - ERB
Library
GTRC
Project File
Other _____

21697-MA

PROGRESS REPORT

TWENTY COPIES REQUIRED

1. ARO PROPOSAL NUMBER: 21697-MA

2. PERIOD COVERED BY REPORT: 1 January 1985 - 30 June 1985

3. TITLE OF PROPOSAL: The Role of Experience in Common-Sense & Expert Problem Solving

4. CONTRACT OR GRANT NUMBER: DAAG29-85-K-0023

5. NAME OF INSTITUTION: Georgia Institute of Technology

6. AUTHORS OF REPORT: Janet L. Kolodner

7. LIST OF MANUSCRIPTS SUBMITTED OR PUBLISHED UNDER ARO SPONSORSHIP DURING THIS REPORTING PERIOD, INCLUDING JOURNAL REFERENCES:

A COMPUTER MODEL OF CASE-BASED REASONING IN PROBLEM SOLVING: An Investigation in the Domain of Dispute Mediation, by Robert Simpson
ARGUMENTS OF PERSUASION: Two Papers, by Katia Sycara-Cyranski
USING EXPERIENCE AS A GUIDE FOR PROBLEM SOLVING, Janet L. Kolodner, Robert L. Simpson

8. SCIENTIFIC PERSONNEL SUPPORTED BY THIS PROJECT AND DEGREES AWARDED DURING THIS REPORTING PERIOD:

Janet Kolodner
Katia Sycara
Robert L. Simpson, who was only indirectly supported by this grant, received his Ph.D. in June, 1985. His dissertation, entitled A COMPUTER MODEL OF CASE-BASED REASONING IN PROBLEM SOLVING: An Investigation in the Domain of Dispute Mediation, is enclosed. His support included computer time and advisory time.

Janet L. Kolodner
Georgia Institute of Technology
Atlanta, GA 30332

BRIEF OUTLINE OF RESEARCH FINDINGS

Our research has taken too directions in the past 6 months. First is the elaboration of our model of experience in problem solving and its implementation in a computer system that resolves disputes. Second is an investigation of strategies for the kinds of persuasion that are necessary in dispute mediation. Several attached papers report this work. Since the paper reporting the first investigation is quite long we summarize it below. The second is summarized in the attached papers.

Our MEDIATOR project resolves common sense disputes based on experience solving previous similar problems. By common-sense disputes, we refer to the kinds people run into from day to day. Children quarrelling over possession of objects, colleagues needing the same resource at the same time, and disputes encountered in reading the newspapers are just a few of the kinds of disputes the program deals with. The MEDIATOR program, developed by Bob Simpson, begins with a semantic memory detailing the kinds of disputes it might encounter (e.g., physical, economic, and political) and a set of common mediation plans (e.g., one cuts the other chooses, split the difference, divide by parts). As it resolves disputes, it builds up an episodic memory organized by the concepts in its semantic memory. During processing, it first attempts recourse to previous experience to resolve a problem, and if no applicable experience is available, it uses default means (based on exhaustive search of alternatives) to resolve the problem. It learns, based on feedback, about the decisions it has made. If feedback is positive, it reinforces its belief that a particular type of plan is appropriate to a particular problem by storing the case and the plan

used to resolve it. When it encounters later problems with features similar to one it has stored in memory, it will be reminded of that case and check to see if the plan used there was appropriate to its new problem. A positive experience may thus provide a shortcut in later problem solving. If feedback is negative, the MEDIATOR tracks down its error, fixes the knowledge that was responsible, and attempts resolution of the problem a second time based on the new knowledge learned during feedback and the corrected knowledge that caused the previous error. When it finally resolves the problem satisfactorily, and stores the entire case in memory, later reminding of that case will (1) allow the problem solver to resolve a later similar problem without making the same mistakes a second time or (2) help the problem solver to figure out what went wrong when a similar failure occurs in the future.

There are several novel aspects to the MEDIATOR project. First, its model of problem solving includes not only the planning part of problem solving, but also problem understanding and failure resolution based on feedback. Case-based reasoning can facilitate reasoning during any of these phases of problem solving.

Second, the analogical transfer process is "demand driven", where demand is provided by the task the problem solver is carrying out. When the problem solver is trying to classify a problem, it is the problem classification of the previous case that it investigates for applicability to the new problem. When it is attempting to derive a skeletal plan, it is the abstract plan from the previous case that it checks for applicability.

Third, the MEDIATOR has a well-articulated long term memory for experience. Problem solving experiences presented to the MEDIATOR are in-

dexed in memory by those features which differentiate it from other experiences stored there. The memory organization is based on MOPs.

A fourth novel feature of the MEDIATOR is in its use of the same problem solving model to both solve domain problems (in this case, to resolve disputes) and to track down and fix failures in reasoning. It is able to do this because it treats both types of problems as first, classification problems, and then, plan instantiation problems. In solving domain problems, it thus seeks to classify disputes it encounters according to whether they are physical, economic, or political disputes during the understanding phase of problem solving. Each of these dispute types "knows" which types of plans are commonly useful to its resolution. Thus, classification allows pointers to potentially applicable canned plans, which must then be refined for the particular problem.

During failure resolution, the MEDIATOR treats the failure it has encountered as its new problem. During the understanding phase of failure resolution (explaining the failure), it attempts to classify the error (as, e.g., a classification error, an elaboration error, a particular kind of elaboration error, a plan refinement error). Each of those error classifications has remediation plans associated with it to fix the faulty knowledge or faulty reasoning rule. It thus fixes its errors by instantiating and refining a plan appropriate to the kind of error it encountered (e.g., one can fix elaboration errors by using an alternate elaboration rule or by asking the value of a feature from the user). In the same way previous experiences can provide shortcuts in problem solving, previous failures can provide shortcuts in error recovery (e.g., the orange dispute above). This method of failure recovery has poten-

tial in domains where the types of failures that may be encountered and ways of recovering from each can be specified.

This fourth feature is the least developed and more investigation is still needed. It is one of our areas for future investigation.

In January, 1985, Dr. Joseph Psotka visted our lab to discuss our projects. We discussed the theoretical framework underlying our endeavor, and he watched a demonstration of our PERSUADER program, which resolves labor/management disputes by generating and proposing an initial settlement, perturbing it based on feedback, and finally generating arguments of persuasion to convince the recalcitrant party of the utility of what is judged the "best" settlement.

21697-MA

PROGRESS REPORT

TWENTY COPIES REQUIRED

1. ARO PROPOSAL NUMBER: 21697-MA

2. PERIOD COVERED BY REPORT: 1 July 1985 - 31 December 1985

3. TITLE OF PROPOSAL: The Role of Experience in Common-Sense & Expert Problem Solving

4. CONTRACT OR GRANT NUMBER: DAAG29-85-K-0033

5. NAME OF INSTITUTION: Georgia Institute of Technology

6. AUTHORS OF REPORT: Janet L. Kolodner

7. LIST OF MANUSCRIPTS SUBMITTED OR PUBLISHED UNDER ARO SPONSORSHIP DURING THIS REPORTING PERIOD, INCLUDING JOURNAL REFERENCES:

Kolodner, J.L., Experiential Processes in Natural Problem Solving, submitted to Artificial Intelligence

Sycara, K., Persuasive Argumentation in Resolution of Collective Bargaining Impasses, Proceedings of the Seventh Annual Conference of The Cognitive Science Society, Aug., 1985.

Sycara, K., Arguments of Persuasion in Labor Mediation, Proceedings of IJCAI-85.

Sycara, K., Precedent-Based Reasoning in Expert Labor Mediation. Thesis Proposal. Technical Report #GIT-ICS-85/22. School of Information and Computer Science, Georgia Institute of Technology, Atlanta, Ga.

8. SCIENTIFIC PERSONNEL SUPPORTED BY THIS PROJECT AND DEGREES AWARDED DURING THIS REPORTING PERIOD:

Janet Kolodner, PI
Katia Sycara, graduate student
T. Rangarajan, graduate student

anet L. Kolodner
ept of Computer Science
eorgia Institute of Technology
tlanta, GA 30332

Brief Outline of Research Findings

Our research has taken two directions in the past six months. First, we have continued basic research looking into a problem solving framework that specifies the role of experience in problem solving. A summary of our findings to date in that area are in the enclosed paper entitled "Experiential Processes in Natural Problem Solving." Much of the work reported in that paper was derived from investigation of common sense dispute mediation in the MEDIATOR program. Of special interest in there are our hypotheses about the ways in which experience can control learning processes. In previous work, we have stated that generalization can be done each time similarities of experience are encountered. Our current hypothesis states that generalizations are appropriate only when a prediction derived from a previous case and used to solve a new problem hold up in problem solving. In this case, those parts of the problems that explain why the prediction was applicable to the new case are generalized. While in our previous formulation, many extra generalizations were made and many co-incidental features were part of generalizations, in this formulation, only potentially useful and explanatory generalizations are formulated. Section 5.2.2 of that paper gives more detail in this area.

Also of interest in deriving that framework is a better-articulated explanation of case-based reasoning processes. While in previous reports (e.g., Simpson, 1985, sent to you previously) we have been able to state in which parts of the problem solving process case-based reasoning was expected, we can now make some more general comments about the case-based reasoning process. We have found that case-based reasoning is used both for hypothesis generation (during problem understanding, planning, and error recovery) and hypothesis evaluation (during planning). We have come up with a set of steps for doing each of these (see sections 5.1.1 and 5.1.2 of the paper referenced above) that make reference to the *demand* or goals of the problem solver. Previous reports of how case-based reasoning was done referred separately to each place in the problem solving cycle where case-based reasoning is useful.

The second direction of our investigation has been to look at experience's roles in problem solving in the domain of labor mediation. Katia Sycara has been carrying on that research. In the past year, she has been defining the domain model of labor mediation. Mediation involves coming up with a solution to a goal dispute, presenting that solution to the parties, taking feedback from the parties, and then either coming up with a new solution based on the feedback or using a persuasive argument to change the mind of a party that does not agree. In the six months previous to this, Katia's concentration was on the persuasive argumentation needed in this last step and the use of previous cases in deriving a solution. We reported on that in the previous semi-annual report. In the past six months, Katia has concentrated on several other things: coming up with a means of deriving a solution when previous experience is NOT available, specifying the domain knowledge necessary to reason about labor contracts, and determining when it is appropriate to derive a new solution and when persuasive argumentation is more appropriate.

The first and last of these topics are particularly interesting. To come up with solutions to them, Katia has derived a utility theory model that measures the potential for agreement to a contract. The model takes into account the relative utility of each of the multiple goals a disputant might have and is used to predict which tradeoffs might be appropriate when everybody's goals cannot be fulfilled. This theory is presented in section 5.2 of the enclosed paper entitled "Precedent-Baed Reasoning in Expert Labor Mediation." Katia has just proposed her Ph.D. dissertation topic, and will be spending the next year doing a further implementation of her work in the PERSUADER program and writing her thesis. Her thesis will have an explanation of the generality and importance of the work she is doing. In particular, she will explain in detail how labor mediation is a real-world instance of dealing with problems that involve multiple competing goals. She will show how a combination of case-based reasoning and the utility theory model she has developed can be used to resolve such problems. Enclosed are several papers she published in last summer's AI and Cognitive Science conferences, her thesis proposal, and some output from her program, the PERSUADER.

21697-MA

PROGRESS REPORT

TWENTY COPIES REQUIRED

1. ARO PROPOSAL NUMBER: 21697-MA

2. PERIOD COVERED BY REPORT: 1 January 1986 - 30 June 1986

3. TITLE OF PROPOSAL: The Role of Experience in Common-Sense & Expert Problem Solving

4. CONTRACT OR GRANT NUMBER: DAAG29-83-K-0023

5. NAME OF INSTITUTION: Georgia Institute of Technology

6. AUTHORS OF REPORT: Janet L. Kolodner

7. LIST OF MANUSCRIPTS SUBMITTED OR PUBLISHED UNDER ARO SPONSORSHIP DURING THIS REPORTING PERIOD, INCLUDING JOURNAL REFERENCES:

Kolodner, J.L., Experiential Processes in Natural Problem Solving, submitted to Cognitive Science

Sycara, K., Utility Theory in Conflict Resolution, submitted to AAAI-86; submitted to a special issue of Annals of Operation Research devoted to the theme "Approaches to Intelligent Decision Support," 1987.

8. SCIENTIFIC PERSONNEL SUPPORTED BY THIS PROJECT AND DEGREES AWARDED DURING THIS REPORTING PERIOD:

Janet L. Kolodner, Associate Professor, PI
Katia Sycara, graduate student
Tom Hinrichs, masters student

Janet L. Kolodner
Department of Computer Science
Georgia Institute of Technology
Atlanta, GA 30332

Brief Outline of Research Findings

Research in the past 6 months has taken two directions: foundational investigations of case-based reasoning and the domain-specific investigation of case-based reasoning in labor mediation. A report of the foundational work will be available at the end of summer, 1986, in a paper to be submitted to *Artificial Intelligence*, co-authored by Janet Kolodner and Robert Simpson. The work to be reported in this paper combines an explanation of what the MEDIATOR does with an analysis, based on the MEDIATOR's strengths and weaknesses, of the processes required in a case-based reasoner.

There are several issues we concentrate on in this analysis. First is an analysis of what gets transferred from a previous case when case-based reasoning is done. The MEDIATOR transferred values all of the time from the previous case to the new one. Certainly, if the details of the cases the reasoner is focussing on are very close, it is appropriate to transfer a value, but in other cases it is more appropriate to *apply the same inference rule used previously* rather than transferring a value. What you get in that case is a real analogy. Example: reminding of the orange dispute when processing the Sinai dispute, attempting to infer goals. By the MEDIATOR's method, the reasoner would throw out the reminding as unacceptable because the objects are so far apart. If it didn't throw the case out, the problem solver would consider whether Israel and Egypt want to eat the Sinai. While it wouldn't actually transfer the value, it's a silly thing to even consider. On the other hand, by the inference method, the problem solver would see that a "default function of object" inference was used to infer the goals in the orange case and apply that to the Sinai, thereby inferring that the countries probably want to set up habitation in the Sinai. That analysis, however, causes some confusion when we look at some of the other domains we have considered. In a medical, domain, for example, it seems appropriate to transfer the values found in the old case (e.g., names of disorders or symptoms to look for), and not the inference rules that derived them. That led us to consider under what circumstances a value is transferred and under what circumstances the inference used previously is applied.

This introduces the second major issue we are analyzing: consistency checks. It is consistency checks that help in determining what ought to be transferred. In the MEDIATOR, because only successful cases were used, there was no need to check for success before doing a transfer. Consistency thus meant that the reasoner needed only to check the preconditions or their equivalent for any value targetted for transfer. This, however, does not allow avoidance of mistakes. When you allow reminding of both successful and failure episodes, the reasoner must check if the value it is trying to transfer was responsible for or changed as a result of reanalysis when a case was a failure, and instead of attempting to transfer the value from the failure episode, it checks to see if the value in the successful instance that goes with it is better (It's actually more complicated, but that's the gist).

We are also trying to differentiate between reinstantiation and real analogy (i.e., a is to b as c is to ?). The MEDIATOR's processes include three case-based operations that derive values to be filled into the representation of the new case: transfer a value, transfer an uninstantiated or partially-instantiated frame, and make an analogy to fill in details. Our current analysis presents some variations on these processes. In particular, the analogical one is done more often and the transfer of a value is done less frequently. More often, a value is compute by transfer of an inference method used previously. In both analyses, however, there seem to be two kinds of mappings that are necessary to derive a value for the new case: mapping over one part of each case and mapping over several (usually two) parts of the cases. The first results in a kind of reinstantiation, while the second is a full-blown analogy. Jerry DeJong of U. of Illinois recently pointed out this difference (at a workshop on similarity and analogy) with respect to several other people's problem solving and learning by analogy systems. His point was that none of those systems were actually doing analogy, that they were only doing reinstantiation. Based on his analysis, we went back and looked at the MEDIATOR, and we believe we will be able to differentiate the two processes from each other and to show when each one is appropriate. It is our complete problem solving model that includes comprehension, planning, and follow-up activities, that will enable us to do that.

Students are currently implementing some of these ideas, though not in the MEDIATOR per se. The MEDIATOR, which we are no longer actively programming, served as a nice testbed for a first pass at implementation and analysis of case-based reasoning processes. Its implementation taught us a lot, and analysis of its functionality may be teaching us even more. At the present time, however, we are implementing these ideas in another interactive problem solving domain. Tom Hinrichs, being paid under this contract since June, 1986, is developing a representational formalism that will allow us to maintain the knowledge needed in making consistency judgements. His formalism will also allow us to have our reasoning systems keep track of several suggestions at once, choosing between them only when it becomes necesary. The MEDIATOR, by contrast, chose one best previous case, made its inferences based on that, and waited until it failed to attempt another line of reasoning. Two other students, paid by other money, but working on the same project, are developing a system that can use this knowledge to make appropriate case-based inferences and building a more general memory traverser than was available in the MEDIATOR.

In her work on the PERSUADER, Katia Sycara continued to concentrate on the utility theory model she developed for deriving solutions when previous experience was not available. She has approximately 6 months of work left to finish her Ph.D. Katia has recently taken a leave of absense for personal reasons and expects to be back in the late fall. We will have a better report of progress on the PERSUADER project in the next report.

Utility Theory in Conflict Resolution

Katia Sycara

School of Information and Computer Science
Georgia Institute of Technology
Atlanta, Georgia 30332
Telephone: (404) 894-5550

ABSTRACT

In this paper we present how multi-attribute utility theory can be incorporated into problem solving situations involving multiple interacting agents, each possessing many goals. Such situations usually involve partial goal satisfaction and persuasion, and have only scantily been described in the AI literature. In this paper we show how utility theory can provide a computational handle to (a) generate a compromise solution that partially satisfies the conflicting goals of the parties, (b) evaluate whether a solution is an improvement on a previously rejected one, and (c) determine the effectiveness of persuasive arguments. Our examples are taken from the domain of labor mediation and are implemented in a computer program, called the PERSUADER.

July 30, 1986

Pages: 13
Science track
Topics:

Automated Reasoning
Cognitive Modeling
Knowledge Representation

Key Words:

Multi-agent Planning
Conflict Resolution
Partial Goal Satisfaction

--

Utility Theory in Conflict Resolution

Katia Sycara

School of Information and Computer Science
Georgia Institute of Technology
Atlanta, Georgia 30332
Telephone: (404) 894-5550

1. Introduction

Perusal of any major newspaper shows that the world is full of conflict situations, for example community disputes, labor-management negotiations, international conflicts. These conflict situations usually involve two or more active agents, each with multiple goals. Such a situation is illustrated in the following example:

> Event 1: The Redhound company is in contract negotiations with its union. The union's demands are a 20 percent wage increase, a 6 percent increase in pensions and seniority as the sole determining factor for promotions and layoffs. The Redhound company's initial position is a 3 percent wage increase, a 1 percent increase in pensions and the criteria for promotions and layoffs to be determined solely by the company. The parties refuse to move from their positions and as a strike deadline approaches, a mediator is called in.
>
> With the mediator's help, the parties shift their positions to the following: the union's demands become 12 percent wage increase, 4 percent increase in pensions; promotions and layoffs to be governed by seniority and ability as co-determinants with higher weight given to seniority. The company's position becomes 6 percent wage increase; 2 percent increase in pensions; the criteria for promotions and layoffs to be determined solely by the company.
>
> After some more discussions, the parties agree to the following proposal of the mediator: 8 percent wage increase, 4 percent increase in pensions; promotions and layoffs to be governed by seniority and ability with ability receiving a higher weight.

The above example illustrates a situation with the following interesting features:

1. There is more than one interacting agent. In the example, the interacting agents are the mediator, the union and the company.*

2. Each of the active agents has more than one goal. These goals are not only different but they interact in certain ways. In Event 1, the union's and company's goals are in conflict in the sense that the union will try to get as much as it can whereas the company will try to give as little as possible.

3. The agents seem to "change their mind" during the course of the interaction. In Event

* For simplicity, we have presented the union and the company as monolithic entities. In the real world factions within the union and company may act as different agents.

1 for example, the union starts with a wage demand of 20 percent increase and after a while it lowers its demand to 12 percent and finally agrees to 8 percent.

4. The parties willingly agree to a settlement that has been proposed by a third party. This settlement gives each party something worse than their initial goals but better than what the other party would be willing to give. In Event 1 for example, the union does not get a 6 percent increase in pensions, but gets something better than the 3 percent increase that the company wanted to give it originally.

The above example illustrates a situation where a solution is sought to resolve the conflicting goals of multiple interacting agents. The bulk of Artificial Intelligence research has ignored problem solving situations where more than one agent was present. However, the presence of multiple agents is an unavoidable reality. People must plan their everyday course of activity taking into consideration the goals and plans of others, which might be in conflict or in concord to their own. A limited Virtually all AI researchers that have looked at multi-agent interactions have assumed that the agents have common or non-conflicting goals [Cammarata83], [Corkill83], [Davis83], [Georgeff84]. The work has thus focused on how these agents can best help each other in achieving their common ends.

On the other hand, there are many situations where the agents have conflicting goals, and where compromise solutions to the conflict would be beneficial to all of them. In an automated factory, for example, robots might be vying for the use of limited resources. An automated battle manager may be required to coordinate a schedule with another automated or human battle manager, while preserving the priorities and goals of its owner. There are situations where a single agent has to plan for conflict resolution among conflicting goals of its "clients". Such an example would be a job shop scheduler [Fox82] that tries to schedule orders among available machines in the best manner. Even in situations where all agents are assumed to have a common goal, sub-goal conflicts may arise. For example, in a distributed problem solving environment, the machines involved in solving a single problem might face conflicts over use of various computational resources. As research in distributed computing progresses, such situations will soon become very common. Thus, research on conflict resolution in multi-agent situations will become increasingly important.

The AI work up to date on conflict resolution of multi-agent goals (e.g., [Genesereth84], [Rosenschein85]) has modeled the problem as one where the agents arrive at a compromise solution through negotiations using game theory. As has been pointed out, [Stevens63], [Bartos74], game theory is not particularly well-suited to model such situations. The structure of a game is represented by the *payoff* matrix -the set of outcomes (payoffs) associated with the various strategies represented by the various rows and columns of the matrix. Each strategy represents a complete sequence of choices appropriate for a particular sequence of contingencies (the opponent's order of choices). If the number of choices available at each move is more than just a few -a most likely case in any realistic situation, the number of strategies implied (rows and columns of the payoff matrix) is enormous. This, in turn, makes actual solution of the game impracticable. Another drawback is the assumption of game theory that each player knows the whole payoff matrix, namely not only his own but also his opponents payoffs. This is clearly not realistic in conflict situations. The game theory formulation cannot accommodate tactics of persuasion and bluff, which are an integral part of negotiations. It assumes, instead that the payoff matrix remains invariant throughout the game [Luce57].

We have chosen to model the conflict resolution as performed by a third impartial agent, the mediator. Mediation has proved its worth in resolving difficult real world conflicts that the agents themselves were unable to resolve through negotiations. In the non-human environment, the role of the mediator is played, for instance by the scheduler of a job shop orders, or by the co-ordinator in a distributed computing environment. There is no single formalism that has been used to model mediation. Our approach involves the use of past cases as well as heuristics and the use of utility theory as the underlying formalism for portraying the parties' preferences. In this paper, we will concentrate on the uses of utility theory in our mediation

model. Our domain of application is labor mediation and our theory is embodied in a computer program called the PERSUADER. The PERSUADER has two general problem solving tasks (a) to construct and propose appropriate compromise settlements to the parties in a labor dispute and (b) to convince the parties to accept a proposed settlement.

2. What is hard in conflict resolution?

Consider the formidable task that confronts a decision maker who has to propose a resolution that takes into consideration the tradeoffs and preferences of multiple agents with multiple conflicting goals. She has to somehow find a settlement that includes "suitable" values of each issue on which both parties will agree. For continuum-valued issues, the choices that such a decision maker has are infinite. For example, the choices that the mediator had in Event 1 were the combination of the infinity of settlements in the range of 3 to 20 percent increase for wages and 1 to 6 percent increase in pensions initially, and afterwards the 6 to 12 percent range for wages and 2 to 4 percent increase in pensions, as well as the range of differences in seniority language. A blind trial and search process is obviously hopeless.

This difficulty arises in every conflict resolution situation. One way for a problem solver to address this difficulty is to subdivide the range of values for each attribute in a sufficiently large number of pieces and consider only the settlements that result from the finite combination of these values. For example, if one subdivides the wage and pension ranges in Event 1 in 6 pieces, the corresponding resulting values in the set of alternatives would be 6, 7, 8, 9, 10, 11, 12 percent increase for wages and 2, 2.33, 2.66, 2.99, 3.32, 3.65, 3.98 for pensions. The finite set of alternative settlements (considering only wages and pensions) is the set formed by all the combinations of the above values. The (mathematically) optimum settlement may not be a member of this set. This is not, however, a serious drawback for problems dealing with human affairs, since people are not optimizers.

Another difficulty is that usually the actors' goals are expressed non-numerically. How could a problem solver compare such goal values? In Event 1, one of the goals is the achievement of a certain language for seniority. A mediator can say that the language "the criteria for promotions and layoffs are to be determined solely by the company" is weaker seniority language than what the union proposed, but she cannot characterize numerically the magnitude of the difference. One solution to this problem is to adopt an arbitrary numerical scale to characterize the non-numerical attributes. This is the solution we have adopted in our implementation for non-economic issues. The chosen scale goes from 0 to 10. In this scale, for example, 10 denotes the strongest seniority language and 0 the weakest.

If a problem solver is to succeed in finding compromise solutions in situations involving many decision makers, he has to have some method of making inferences about the ways the decision makers evaluate alternative solutions and make choices. The method used by the problem solver should also allow him to take into consideration possible tradeoffs that the decision makers would be willing to accept. Utility theory provides such a methodology. Utility theory is the theory that models the process through which a decision maker evaluates a set of alternatives, so that he can choose the best one. It has also been used in aiding a decision maker to structure his problem in such a way that evaluation of the alternatives is easily accomplished [Whit74], [Keeney75]. In this paper, we concentrate on the novel ways that utility theory can be exploited in problem solving. In our model, utility theory is used by the problem solver to (1) generate potentially acceptable solutions to be proposed to the parties, (2) measure the quality of a modification to a rejected settlement, and (3) determine the effectiveness of persuasive argumentation.

3. Utility Theory in brief*

* For an extended treatment of utility theory see [Keeney76].

The concept of *utility* is the basis for selecting among future alternatives and for evaluating past actions. Each time a house is bought, or the choice of a job has to be made, or any other form of action has to be taken, some form of assessment of utility of the various alternatives for the decision maker is used in order to make the decision. Each alternative is evaluated in terms of a number of attributes that the decision maker considers important. In bying a house, for example, some of the relevant attributes are cost, distance from work, safety of neighborhood, quietness. Each prospective house is evaluated on each one of these attributes. It is vary rare indeed, that a particular alternative will have the best rating on all the attributes under consideration. Thus, a decision maker must have a way of comparing alternatives with varying attribute values, in order to pick the one that offers him the maximum overall utility, or satisfaction. This is not easy, since the individual utilities associated with the attributes are not linear in general. For example, suppose that the safety of a neighborhood was rated on a scale of 0 (totally unsafe) to 10 (totally safe) points. Further suppose that a mother of small children were to be asked about the utility of this attribute on an (arbitrary) scale from 0 to 100% satisfaction. It is very plausible that she would give the following ratings: zero percent satisfaction for safety in the range of 0 to 4 points, 25% for a safety rating of 5, 75% for a rating of 9 and 100% for a rating of 10. This is obviously a non-linear relationship. Moreover, a different decision maker, a Mafia tough for instance, if asked to rate his satisfaction with the safety attribute on a 0 to 100% scale, would give quite different ratings than the mother. Thus, not only is the utility associated with an attribute nonlinear, but it also varies with the decision maker.

Another difficulty that arises in comparing alternatives is that a a decision maker must accept lower values on some attributes in order to get higher values on others. In other words, he must make trade-offs. Because the measurement scales of the attributes are in general incommensurate, one unit of one attribute does not have the same utility as one unit of another attribute. To continue with the house buying example, even if quietness is measured on a scale of 0 (totally noisy) to 10 (totally quiet), one unit of safety is probably not equivalent to one unit of quietness. In other words, the mother decision maker would not be indifferent (i.e. derive the same satisfaction) between two houses with the same cost, same distance from work but one with 5 units of safety and 4 of quietness and the other with 5 units of quietness and 4 of safety. Thus, a decision maker must know how many units of one attribute he is willing to give up in order to gain one unit of another attribute. The individual utility relations as well as the tradeoff values for the various attributes constitute the *preference structure* of a decision maker. This preference structure potentially varies with each decision maker. Utility theory provides a methodology through which a decision maker's preference structure can be identified, so that utility assessments of alternatives can be made.

We briefly describe formally the general problem that utility theory addresses. It is the following: how should a decision maker decide to choose an act β out of a set **A** of action options, such that he will be happiest with the consequence/payoff associated with this choice? We give an abstract formulation of this problem. The action options are $A_1,...,A_m$. There are a set of attributes of concern $X_1,...,X_j,...,X_n$, and each option β can be evaluated on each of these attributes to get n indices of value $X_1(\beta),...X_n(\beta)$. In labor mediation, for example, the action options are the various possible contract settlements and the attributes are the contract issues, e.g., wages, seniority, pensions. Let the evaluation of option A_i on attribute X_j be given by the number x_{ij} for $i=1,2,...,m$ and $j=1,2,...n$. Thus, option A_i can be identified with a vector consequence $x_i = (x_{i1}, x_{i2},..., x_{ij},..., x_{in})$. Thus, a comparison between two options involves comparisons between two n-tuples. For example, a contract γ with a 47 cents increase in wages and a 6 cents increase in pension benefits would be expressed in the above notation as $\gamma = (47, 6)$, assuming that the ordering of the attributes is (wages, pensions).

Since attributes X_i and X_j may in general be measured in different units, it is meaningless to compare elements x_i and x_j ($i \neq j$) of an n-valued payoff. Thus, for each option β, we would want to find an index that combines the n-valued payoff $X_1(\beta), X_2(\beta),....X_n(\beta)$ into a *scalar* value-function v that expresses the preferability of option β for the decision maker. This function is called *value* or *utility function*. Given v, the decision maker's problem is to choose β in **A**

such that v is maximized. In the case where the chosen action has to satisfy the goals of more than one decision maker, as in labor mediation, the alternative that maximizes the combined payoff of the decision makers is selected.

The utility function v, defined on the consequence space has the property that

$$v(x_1,....,x_n) \geq v(x_1',...,x_n') \; iff \; (x_1,...,x_n) \geq (x_1', \ldots, x_n') \tag{1}$$

where the symbol $\geq$ reads "preferred or indifferent to", and "iff" means "if and only if".

It would tremendously simplify the calculations, if we could find a function, call it f, with a simple form such that

$$v(x_1,x_2,...,x_n) = f[v_1(x_1),v_2(x_2),...,v_n(x_n)], \tag{2}$$

where v_i designates a value/utility function over the single attribute X_i. We are interested in conditions when expression (2) holds. The simplest and most useful form that expression (2) can take is the *additive* form, namely

$$v(x_1,...,x_n) = w_1v_1(x_1) + w_2v_2(x_2) + \cdots + w_nv_n(x_n) \tag{3}$$

where w_i designates the weight/importance that a decision maker attaches to each attribute.

One nice property that an additive function has is that it is compensatory in the sense that an increase in the utility of one attribute can compensate for a decrease in the utility of any other attribute. Thus, such a function models well the tradeoffs that a decision maker is willing to consider. Additive functions are the ones most frequently used in practice [Johnson77], [Keeney76].

3.1. Deriving Utilities

If a decision maker has available the utility functions associated with each attribute under consideration, he can take their weighted sum to arrive at an overall utility function for all attributes of interest (cf equ. (3)). This function maps the individual utility values associated with a particular alternative to a single number, the satisfaction of the decision maker with that alternative. To make his final choice, the decision maker selects the alternative that maximizes the overall utility function. As we saw in the previous section, because of the nonlinearity of the relation between the individual utilities and the associated attributes, and the noncommensurability of attribute scales, the assessment of a decision maker's utilities is not an easy problem. To obtain the utility curves of the parties, a problem solver can (a) follow an assessment procedure that elicits the decision maker's utilities via direct questioning, (b) retrieve the utility curves of similar parties from past successful problem solving episodes or (c) hypothesize the shape of the curves from knowledge of domain-specific factors.

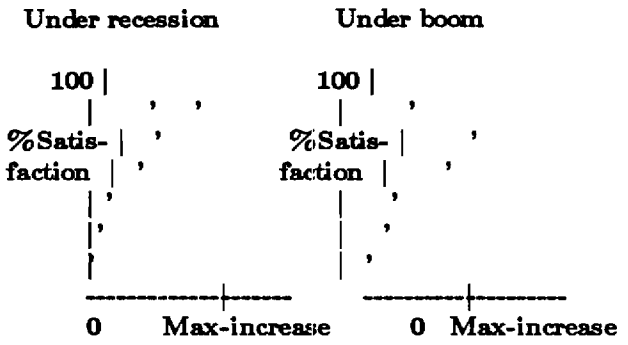
There is a variety of utility assessment procedures that directly question the decision maker. Each procedure indicates the kind of questions that can be asked of the decision maker to elicit individual utilities and some guidance as to how the individual utilities are to be combined to obtain the overall utility function. Space limitations prevent us from presenting such procedures, especially since they are well documented in the decision analysis literature (for a survey of these procedures, see [Johnson77]). Though these utility assessment procedures seem to elicit accurate utilities, they are time consuming and in many cases impractical (e.g., they presuppose trust on the part of the decision maker towards the questioner).

Retrieval from memory of the utility curves of similar parties is another way to obtain them, and is the preferred method in our model. In our implementation, the curves are stored as part of the profile frames of the agents whose goals are in conflict. We assume that the utility curves of similar agents for a particular attribute will have the same functional form. This assumption is supported by various experimental studies (e.g., [Swalm66], [Spetzler68]). But what makes agents similar? The answer depends on the domain under investigation. The criteria for similarity of disputants in the domain of labor mediation that the PERSUADER uses are similarity of industry, similarity of geographical location, same international union. These

criteria are reasonable because they reflect the economic realities of the negotiation situation. For example, two paper mills in Georgia are assumed to have the same utility curves for the same contract issues.

Hypothesizing the utility curves of the parties is a third general method that a problem solver can use to obtain utilities. This method relies heavily on domain-specific heuristics. In labor mediation, the factors that are used in the heuristics are the state of the economy in the industry, the unemployment rate for the bargain unit's job classification in the area, and the structure of the bargaining unit (e.g., proportion of skilled vs. unskilled workers, young vs. old). The following figure shows how the factor of economic boom or recession impacts the health-benefits curve of a union.

POSSIBLE UNION UTILITY CURVES FOR HEALTH-BENEFITS



Notice that in both cases the union will not be satisfied at all if it is not given any health benefits increases, and it will be 100% satisfied if it is given the maximum increase. Under boom, the union will be less than 50% satisfied if it is given an increase of magnitude $[\frac{1}{2}Max-increase]$, whereas under recession, it will be more than 50% satisfied if the conceded increase is $[\frac{1}{2}Max-increase]$. Thus, the two curves in the figure reflect qualitatively the realities of a union's satisfaction under two different economic conditions. It is reasonable, therefore for the mediator to hypothesize the shape of these curves. Elementary calculus gives analytic expressions for these two curves. The utility curve under recession can be expressed as

$$y(x) = \frac{-100}{Max^2}(x-Max)^2 + 100$$

and the utility curve under boom can be expressed as

$$y(x) = \frac{100}{Max^2}x^2$$

where, for notational simplicity, Max-increase is denoted by Max.

Hypothesized utility curves and those derived from the utility curves of similar parties are not as accurate as the ones derived from direct assessment techniques. This, however, is not a great disadvantage in our model, since these curves are used to propose an initial solution and get modified in the course of problem solving via persuasive argumentation.

In the previous paragraphs we have briefly presented three general methods for obtaining the utilities of a decision maker. The PERSUADER uses the method of retrieving the curves of similar decision makers that have been encountered in previous problem solving episodes. If no such past experience is available, the PERSUADER hypothesizes the utility curves by using a set of domain-specific heuristics to select the appropriate curves from a number of curves that it knows about. Once the utility curves have been obtained for the contract issues under

negotiation in the present case by either method, the PERSUADER constructs the overall utility function for each party by asking the parties directly for the weights they attach to these issues and forming the weighted sum (cf eq. 3) of the retrieved curves. The PERSUADER uses the utility curves of the parties constructed thusly in three ways during the problem solving process: (1) to suggest a potentially acceptable compromise solution, (2) to evaluate the quality of the modification to a rejected settlement, and (3) to determine the effectiveness of persuasive argumentation.

3.2. Utility theory in generating a compromise solution

As has been illustrated in section 2, the task of proposing a potentially acceptable resolution that takes into consideration the tradeoffs and preferences of multiple agents with multiple conflicting goals is a hard one. How can utility theory help? A utility function models a decision maker's preferences so that he can evaluate a set of multi-attribute alternatives and select the best one. In conflict resolution situations the alternative that would be the most preferable for one agent would most likely be the least preferable for another, since their goals are in conflict. Thus, a third party problem solver is faced with the problem of how to select a compromise solution that will be potentially acceptable to all parties. We assume that the parties are reasonable enough to know that they cannot get the settlement that is most preferable to them, since somebody else is bound to object to it. We have considered two heuristic criteria that seem reasonable and can guide the problem solver in selecting the "best" compromise solution: (1) maximizing the *joint payoff**, and (2) minimizing the *payoff difference* of the parties.

To apply the first criterion, one can proceed as follows: By range subdivision of each of the attribute values and combination of the resulting values, (see section 2), a finite set of alternatives is constructed. For simplicity, let us consider the case where there are two issues under consideration. For example, in Event 1, considering only wages and pensions as the issues, one of the alternatives would be (6, 2), namely a 6% increase in wages and 2% increase in pensions (assuming the range subdivision of section 2). Adapting eq. (3) for two issues, the company's (and union's) utility curves can be expressed by the general formula

$$v(x_1,x_2) = \alpha v_1(x_1) + (1-\alpha) v_2(x_2), \tag{4}$$

where α and $(1-\alpha)$ are the weights and $v_1(x_1)$ and $v_2(x_2)$ the utility curves for wages and pensions for each respective party. Thus, $v(6,2)$ can be calculated for each party. The joint payoff of the parties is given by the general formula

$$U(x_1,x_2) = u_1(x_1,x_2) + u_2(x_1,x_2), \tag{5}$$

where $u_1(x_1,x_2)$ is the company's utility curve and $u_2(x_1,x_2)$ for settlement (x_1,x_2). The joint payoff of the parties for each settlement under consideration can be calculated using eq. (5). Then, the alternative that gives the maximum of these values is selected and proposed.

Another possible criterion could be to select the most fair solution, namely the one with the smallest difference in the parties' payoffs. This is done as follows: Once the parties payoffs for each alternative have been calculated using eq. (4), the difference

$$U_d(x_1,x_2) = \left| u_1(x_1,x_2) - u_2(x_1,x_2) \right| \tag{6}$$

is calculated (assuming u_1 is the utility curve of the company and u_2 of the union). The alternative that minimizes this difference is selected.

In order to decide which criterion the PERSUADER would use, we ran a few examples using each one. Maximizing the joint payoff very often gave contracts with quite unequal utilities for the parties. In those experiments, the payoff of one party would be so low as to practically guarantee rejection of the settlement by that party. On the other hand, minimizing the difference can lead to absurd results. For instance, this criterion would not be able to

* Maximizing the joint payoff has been suggested in [Raiffa82].

differentiate between alternatives one of which gives both parties a payoff of 40, and another that gives both parties payoff 70 (since in both cases the payoff difference is 0). Hence, we chose to combine the two criteria and select the alternative that minimizes the difference and maximizes the joint payoff. This is done by computing the joint payoff (eq. 5) and the payoff difference (eq. 6) for each alternative, taking the difference of these two and selecting the alternative that maximizes this difference.

3.3. Utility Theory in Finding a Better Solution

No matter how the utility functions of the parties have been obtained, they are not completely accurate [Johnson77], [Shepard64]. Moreover, the preference structures of the agents can change during problem solving [Bartos74], [Swingle70]. Thus, a suggested compromise solution may be rejected by either or both parties. A problem solver needs to be able to suggest another solution that will be no worse than the rejected one in the sense that it will have at least the same chance of been accepted. To do this, a problem solver has to have some criterion that progress is being made every time a new solution is to be proposed. The parties' payoffs give such a criterion. A problem solver employs various plans to create a new solution by modifying the rejected one [Sycara85c]. These plans are domain dependent and the result of their application on the rejected solution is predictable. For example, if there is a high turnover of workers in a company, a mediator can infer that they would not be very interested in strong seniority language, and thus she can employ a plan to further weaken the seniority language in a situation where the company has rejected a suggested settlement.

To see how the parties' payoff can be used as a criterion of whether a modified settlement has improved its chance of acceptability, consider the following example: Suppose that a proposed contract with 40 cents increase in wages and 10 cents increase in pensions and with payoffs 52% for the company and 62% for the union, is rejected by the company. The mediator proposes a 3 cents pension reduction resulting in the contract (40, 7). The mediator calculates the payoffs of the parties for the contract (40,7) by using eq. (4). Suppose these payoffs are 61% for the company (an increase of 8%) and 58% for the union (a decrease of 4%). The criterion that is used to decide whether to suggest the modified contract is that it increase the rejecting party's payoff by a greater amount than it might decrease the payoff of the party that had accepted the previously proposed contract.* In the above example, the contract (40, 7) will be suggested. Thus, a problem solver does not waste time in proposing solutions that are inferior to rejected ones. The incremental solution improvement process is akin to *hill climbing* and the above criterion affords the test for proceeding. In our implementation, we assume that a solution that affords both parties a payoff greater than or equal to 70%** will be accepted by the parties. The parties might of course choose to accept a settlement that gives less than 70% payoff.

3.4. Utility theory in persuasion

In conflict resolution, a problem solver uses persuasive argumentation to convince a party that rejected a settlement to accept it, or to narrow the parties' differences with respect to the issues' values by convincing a party to accept a lower value than the one he demanded. To accomplish this, a problem solver needs a computational handle on the notion of "convincing" someone to accept a settlement/value that was previously unacceptable to him. The notion of payoff supplies such a handle. "Convincing" somebody can be modeled as increasing the payoff that the settlement/value gives him. Hence, the task of a persuader can be viewed as finding the most effective argument that will increase a party's payoff with respect to a

* In most conflict situations if a resolution that was acceptable to one party is modified in favor of the opposing party, the resulting resolution will give the party that had accepted a smaller payoff than the previous resolution.

** This number has been checked for approximate accuracy by practicing mediators. The reason that 100% satisfaction with a resolution is not necessary is that the parties are assumed to be reasonable, in the sense that they know that they need to compromise.

settlement/value.***

Equation (3) shows that there are two ways to increase a party's payoff: (a) by changing the weight/importance the party attaches to an issue, and (b) by changing the value of an issue. These two ways can be viewed as a persuader's argumentation goals. In the rest of this section we give a brief description of how a utility-derived formulation can guide a persuader in the selection of argumentation strategies and particular arguments to achieve the argumentation goals (for a more detailed treatment, see [Sycara85a, Sycara85b]).

For simplicity, let us consider the case where there are two issues under consideration. Then, eq. (3) becomes

$$v(x_1,x_2) = \alpha v_1(x_1) + (1-\alpha) v_2(x_2) \tag{7}$$

where x_1 and x_2 are the values of the two contract issues, v_1 and v_2 are the individual utility curves associated with the issues, and α and $(1-\alpha)$ are the weights (or relative importance) that the party accords the contract issues. If the weights are changed, the payoff also changes. This reflects the intuitive notion that satisfaction with a thing is a function not only of the intrinsic value of the thing but also of the importance that we attach to it, our view of it. The sign of the partial derivative of v with respect to α indicates the direction of change of v.

$$\frac{\partial v}{\partial \alpha} = v_1 - v_2; \text{ so if } v_1 \geq v_2, \text{ to increase } v \text{ increase } \alpha \tag{8a}$$

$$\text{if } v_1 < v_2, \text{ to increase } v \text{ decrease } \alpha. \tag{8b}$$

Thus, relations (8a) and (8b) show how the weights can be changed, so that an increase in payoff will result. Moreover, (8a) and (8b) show that the change in the weights of one party can be carried out independently of any weight changes for the other party. Since there are more than one issues involved, a persuader needs to find out (a) which issue's importance she should try to change, and (b) in what direction (increase or decrease). Equations (8a) and (8b) give us a criterion for answering these two questions. A persuader has access to the parties' utility curves as well as the importance that the parties attach to the various issues. Thus, when a party has rejected a proposed settlement, a persuader can check the relationship ($\geq$ or $<$) of the utility curves for the values of the issues in the proposed resolution. Then, she can use equation (8a) or (8b) to decide which issue's importance to increase or decrease. This is the procedure the PERSUADER uses. In the PERSUADER, arguments are accessed with respect to (a) the issue to which they pertain and (b) whether they increase or decrease the issue's importance [Sycara85c].

Another argumentation goal of a persuader is to change the assessment of the value of the issue under discussion in the persuadee's eyes. In the mediation domain, "the issue under discussion" is a contract issue, and its value is the monetary value of the contract issue. In the utility theory model, changing a party's assessment of the value of an issue is equivalent to changing the party's satisfaction curve at that value, thus raising the party's payoff. Consider, for instance the situation where a company assesses an increase of 20 cents an hour in wages as "too high". In the utility theory formulation, this can be translated as $vc(20)=35$ (i.e., the satisfaction that the company would derive if it were to give a wage increase of 20 cents per hour would be 35%). Convincing the company that this increase is not so high, is equivalent to raising its satisfaction (or in mathematical notation, $vc(20)>35$). In [Sycara85c], we have presented strategies of persuasion that a persuader can employ to achieve her argumentation goals.

It would be an ineffective persuader, however, if one were unable to recognize when argumentation is useless. This happens when a party's payoff is so low that no amount of persuasion can convince him to change his position. Since a persuader can calculate the parties' payoffs that a proposed settlement gives them, she can decide whether argumentation is effective. In

*** Modeling a persuader's task thus, can be applied to general situations of persuasion, not just conflict resolution. In the general case, a persuader needs to convince somebody of a proposition, not just a settlement.

our implementation, we consider that if a proposed settlement gives a party a payoff less than 50%*, it is useless to argue and it is better to try to form another potentially acceptable settlement.

4. An example from the PERSUADER

The PERSUADER is a computer program that, presented with an impasse in labor negotiations between a company and a union, discovers and proposes reasonable contract settlements and persuades the parties to accept the proposed settlement. It uses *precedent-based reasoning* [Sycara85c], namely the retrieval of similar contracts from its memory of past experiences, to use as a starting point for its reasoning. We present the following example, where the PERSUADER resolves a collective bargaining dispute involving the VARTA transit company and the bus drivers' union. The user's feedback is in boldface. The current impasse in negotiations is presented. The union wants a 12% wage increase and a 6% increase in pensions, while the company proposes a 3% increase in wages and no pension increase.

(intro VARTA-dispute)
The present contract negotiation is VARTA company contract negotiations
The negotiations are at an impasse, which I will try to resolve
Looking to propose a contract that is acceptable
to #<M-LOCAL 16271553> (VARTA union)
and #<M-TRANSIT-COMPANY 16274557> (VARTA company)
The impasse resulted from the following:
VARTA-union has made the demands
#<M-WAGE-GOAL 16272354> (12% wage increase)
#<M-PENSION-GOAL 16272565> (6% increase in pensions)
VARTA company has responded with the proposals
#<M-WAGE-GOAL 16272377> (3% wage increase)
#<M-PENSION-GOAL 16272612> (no increase in pensions)

Searching memory for current competitors' contracts...
Having found current competitors' contracts...
The contract I am considering is the current contract
for #<M-TRANSIT-COMPANY 16271555> (NARTA company)

The PERSUADER checks the retrieved contract to see whether wages and pensions, the issues in the current dispute were negotiated in the case of the NARTA company. Indeed, they were. The frame representing the NARTA company contains its utility curves for wages and pensions. The frame representing NARTA's union correspondingly contains its utility curves for wages and pensions. NARTA company's wage utility curve is (in algebraic representation) $w_c(x_1) = \frac{100}{(B_1 - A_1)^2}(x_1 - B_1)^2$, where A_1 and B_1 are the variables representing the endpoints of the range for wage values. NARTA company's pension utility curve is $p_c(x_2) = \frac{100}{(A_2 - B_2)}(x_2 - B_2)$, where A_2 and B_2 are the variables representing the endpoints of the range for the pension values. Correspondingly, the NARTA union's utility curve for wages is $w_u(x_1) = \frac{-100}{(B_1 - A_1)^2}(x_1 - B_1)^2 + 100$, where A_1 and B_1 have the same meaning as above, and its pension utility curve is $p_u(x_2) = \frac{100}{(B_2 - A_2)}(x_2 - A_2)$, where A_2 and B_2 have the same meaning as above.

* We have checked this number with practicing mediators who told us that it is about right.

Subdividing the range [12, 3] of #<M-WAGE-GOAL 16322757>
into 4 pieces...
Subdividing the range [6, 0] of #<M-PENSION-GOAL 16323031>
into 4 pieces...
The considered contracts are formed by combining
all the endpoint values derived from the range subdivision.
There are two issues in dispute...
Combine the values of these two issues.
There are 25 contracts under consideration.

Need to find the weights the parties attach to the issues.

*** THE MEDIATOR TALKS PRIVATELY TO #<M-LOCAL 16271553> ***
What would you #<M-LOCAL 16271553> say is the relative importance
on a scale of 0 to 1 of #<M-PENSION-GOAL 16323031>
with respect to #<M-WAGE-GOAL 16322757> for you? **.7**

*** THE MEDIATOR TALKS PRIVATELY TO #<M-TRANSIT-COMPANY 16274557> ***
What would you #<M-TRANSIT-COMPANY 16274557>
say is the relative importance
on a scale of 0 to 1 of #<M-PENSION-GOAL 16323031>
with respect to #<M-WAGE-GOAL 16322757> for you? **.4**

Using company #<M-TRANSIT-COMPANY 16271555> (NARTA company)
utility curves as curves of #<M-TRANSIT-COMPANY 16274557>
Using #<M-LOCAL 16273551> (NARTA union)
utility curves as curves of #<M-LOCAL 16271553>

Calculating company's utility for each of the 25 contracts
using weighted sum of company utility curves for the issues.
Calculating union's utility for each of the 25 contracts
using weighted sum of union utility curves for the issues.

Checking to see whether any contract gives both parties payoff > 70...
No contract gives both parties payoff > 70.

Calculating joint payoff for each of the 25 contracts
using simple sum of parties' payoffs.
Calculating difference payoff for each of the 25 contracts
using simple difference of parties' payoffs.

Calculating difference between the joint payoff and the difference payoff
for each of the 25 contracts.
Selecting contract (6.0, 4.0) corresponding to maximum difference
since this contract minimizes the difference and maximizes the joint payoff.

The contract #<M-CONTRACT 16276047>
with 6.0 change in the value of #<M-WAGE-GOAL 16322757>
and 4.0 change in the value of #<M-PENSION-GOAL 16323031>
minimizes the difference and maximizes the joint payoff
Contract #<M-CONTRACT 16276047>
will be proposed to solve the impasse.

The proposed contract has a 6% wage increase and a 4% increase in pension benefits. Though the PERSUADER calculated that the company cannot quite afford this economic package, it is nevertheless proposed since the PERSUADER knows of a plan to pass the extra cost to the consumer. The PERSUADER proposes this settlement to both sides.

*** THE MEDIATOR TALKS PRIVATELY TO THE PARTIES ***

Do you #<M-LOCAL 16271553> accept #<M-CONTRACT 16276047>
as a way to solve the impasse ? **yes**

Do you #<M-TRANSIT-COMPANY 16274557> accept #<M-CONTRACT 16276047>
as a way to solve the impasse ? **no**

The company rejects the proposed contract. The PERSUADER's goal now is to convince the disagreeing party to agree. It finds, however that the rejecting party's payoff for this contract is 45% and, considering it too low decides not to try to use persuasive arguments to increase it. It now considers another plan for an acceptable settlement. First, the PERSUADER checks to see whether the plan's preconditions are satisfied. Then it checks to see whether the plan's application will result in an improved solution.

Looking at the plan called "try to reduce the cost of pensions"

With respect to pensions,
since the bargain unit consists mainly of young workers
a reduction in pension cost seems acceptable

The settlement has still 6% wage increase but only 1% increase in pension benefits. The PERSUADER checks the parties' payoffs. The company's payoff increased by 20% (from 45 to 65) and the union's payoff decreased by 12% (from 67 to 55). So, the new settlement is proposed.

The contract #<M-CONTRACT 16276047> which resulted
from the plan #<M-REDUCE-PENSIONS 16324663>
will be proposed to solve the impasse.

*** THE MEDIATOR TALKS PRIVATELY TO THE PARTIES ***

Do you #<M-LOCAL 16271553> accept #<M-CONTRACT 16276047>
as a way to solve the impasse ? **no**

Do you #<M-TRANSIT-COMPANY 16274557> accept #<M-CONTRACT 16276047>
as a way to solve the impasse ? **yes**

*** THE MEDIATOR'S SOLILOQUY ***

Since the #<M-LOCAL 16271553> rejected the contract
I need to find the weights #<M-LOCAL 16271553>
attaches to the issues

The PERSUADER asks for the weights of the rejecting party again to check whether they have changed. In this case, the union's weights have not changed. Hence, the payoff remains at 55%

The #<M-LOCAL 16271553> 's payoff for the contract #<M-CONTRACT 16276047>
with 1.0 change in the value of #<M-PENSION-GOAL 16323031>
and 6.0 change in the value of #<M-WAGE-GOAL 16322757>
is (55.0)
Since #<M-LOCAL 16271553> rejected the contract,
the payoff needs to be increased, if appropriate

Since the value in utility curve of #<M-WAGE-GOAL 16322757>
is greater than the value in utility curve for #<M-PENSION-GOAL 16323031>
try to find an argument to increase the weight of wages
or, equivalently, decrease the weight of pensions.

The PERSUADER observes that it is unusual for a union with a majority of young members to give such high importance to pensions and forms the hypothesis that the great weight to pension increases may be due to a goal set by the international union. The program checks and verifies this hypothesis in the current case.

Then, the PERSUADER searches memory for appropriate arguments, namely arguments that have been used in the past to convince the rejecting party (the union) that the importance it attaches to pensions is too high and it evaluates whether the argument would be applicable in the present case.

*** THE MEDIATOR TALKS PRIVATELY TO #<M-LOCAL 16271553> ***

It is o.k. for the international union
to have a high pension goal,
but your workers are mostly young
so, they won't be disappointed
to receive lower pension benefits.

Do you agree ? **yes**

Do you #<M-LOCAL 16271553> accept #<M-CONTRACT 16276047>
as a way to solve the impasse ? **yes**

The mediator accepts congratulations
---

5. Summary

In this paper, we have presented how utility theory can be incorporated in problem solving in situations involving multiple agents with multiple conflicting goals. Utility theory is used for (a) generation of solutions to be proposed to the parties, (b) measuring the quality of a modification to a rejected solution, and (c) measuring the effectiveness of persuasive argumentation.

References

[Bartos74] Bartos, O., *Process and Outcome of Negotiations*, Columbia University Press, New York and London, 1974.

[Cammarata83] Cammarata, S., McArthur, D., and Steeb, R., "Strategies of cooperation in distributed problem solving", *IJCAI-83*, pp. 767-770, Karlsruhe, West Germany, 1983.

[Corkill83] Corkill, D. D., and Lesser, V. R., "The use of meta-level control for coordination in a distributed problem solving network", *IJCAI-83*, pp. 748-756, Karlsruhe, West Germany, 1983.

[Davis83] Davis, R., and Smith R. G., "Negotiation as a metaphor for distributed problem solving", *Artificial Intelligence*, vol. 20, pp. 63-100, 1983.

[Fox82] Fox, M. S., Allen, B., and Strohm., G. "Job-Shop scheduling: An investigation in Constraint-Directed Reasoning", *AAAI-82*, pp. 155-158, Pittsburg, PN., August 1982.

[Genesereth84] Genesereth, M. R., Ginsberg, M. L., and Rosenschein, J. S., "Cooperation without Communication", *HPP Report 84-36*, 1984.

[Georgeff84] Georgeff, M. A., "Theory of action for multi-agent planning", *AAAI-84*, pp. 121-125, Austin, Texas, 1984.

[Johnson77] Johnson, E. M., and Huber, G. P., "The Technology of Utility Assessment", *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-7, no. 5, pp. 311-325, May 1977.

[Keeney75] Keeney, R. L., and Nair, K., "Decision analysis for the siting of nuclear power plants-The relevance of multiattribute utility theory", *Proceedings of the IEEE*, vol. 63, pp. 494-500, 1975.

[Keeney76] Keeney, R. L., and Raiffa, H., *Decisions with Multiple Objectives*, John Wiley and Sons, New York, 1976.

[Luce57] Luce, R. D., and Raiffa, H., *Games and Decisions: Introduction and Critical Survey*, John Wiley and Sons, New York, 1957.

[Raiffa82] Raiffa, H., *The Art and Science of Negotiation*, The Harvard University Press, Cambridge, Mass., 1982.

[Rosenschein85] Rosenschein, J. S., and Genesereth, M. R., "Deals Among Rational Agents", *IJCAI-85*, vol. 1, pp. 91-99, Los Angeles, Ca., August 1985.

[Shepard64] Shepard, R. N., "On subjectively optimal selection among multiattributed alternatives", *Human Judgement and Optimality*, Shelley, M. W., and Bryan, G. L., Eds. Wiley and Sons, New York, 1964.

[Spetzler68] Spetzler, C. S., "The development of a corporate risk policy for capital investment decisions", *IEEE Transactions on Systems, Science and Cybernetics*, SSC-4, pp. 279-300, 1968.

[Stevens63] Stevens, C. M., *Strategy and Collective Bargaining Negotiation*, McGraw-Hill Book

Company Inc., 1963.

[Swalm66] Swalm, R. O. "Utility theory-insights into risk taking", *Harvard Business Review*, vol. 44, pp. 123-136, 1966

[Swingle70] Swingle, P., Ed. *The Structure of Conflict*, Academic Press, New York, 1970.

[Sycara85a] Sycara-Cyranski, K. "Arguments of Persuasion in Labour Mediation", *IJCAI-85*, vol. 1, pp. 294-296, Los Angeles, Ca., 1985.

[Sycara85b] Sycara-Cyranski, K. "Persuasive Argumentation in Resolution of Collective Bargaining Impasses", *Proceedings of the Seventh Annual Conference of The Cognitive Science Society*, pp. 356-360., Irvine, Ca., 1985

[Sycara85c] Sycara-Cyranski, K. "Precedent-based Reasoning in Expert Labor Mediation", *Georgia Institute of Technology Report Git-ICS-85/22*, Atlanta, Ga., 1985

[Whit74] Whitmore, G. A., and Cavadias, G. S., "Experimental determination of community preferences for water quality-cost alternatives", *Decision Sciences*, vol. 5, pp. 614-631, 1974.

[Kolodner84] Kolodner, J. L., *Retrieval and Organizational Strategies in Conceptual Memory: A Computer Model*, Lawrence Erlbaum, Hillsdale, NJ., 1984

[Kolodner85] Kolodner, J. L., Simpson, R. L., and Sycara-Cyranski, K. "A process model of case-based reasoning in problem solving", /fIIJCAI-85, pp. 284-290, Los Angeles, Ca., 1985

G-2

21697-MA

PROGRESS REPORT

TWENTY COPIES REQUIRED

ARO PROPOSAL NUMBER: 21697-MA

PERIOD COVERED BY REPORT: 1 July 1986 - 31 December 1986

TITLE OF PROPOSAL: The Role of Experience in Common-Sense & Expert Problem Solving

CONTRACT OR GRANT NUMBER: DAAG29-85-K-0023

NAME OF INSTITUTION: Georgia Institute of Technology

AUTHORS OF REPORT: Janet L. Kolodner

LIST OF MANUSCRIPTS SUBMITTED OR PUBLISHED UNDER ARO SPONSORSHIP DURING THIS REPORTING PERIOD, INCLUDING JOURNAL REFERENCES:

Sycara, K. Finding Creative Solutions in Adversarial Impasses. Submitted to AAAI-87.

Sycara, K. Utility Theory in Conflict Resolution. Accepted for special issue of Annals of Operations Research, entitled "Intelligent Decision Support Systems."

Kolodner, J.L. (1987). Capitalizing on Failure Through Case-Based Inference. Submitted to AAAI-87.

SCIENTIFIC PERSONNEL SUPPORTED BY THIS PROJECT AND DEGREES AWARDED DURING THIS REPORTING PERIOD:

Katia Sycara-Cyranski, Ph.D. student
Tom Hinrichs, Ph.D. student, summer, 1986

et L. Kolodner
artment of Computer Science
rgia Institute of Technology
anta, GA 30332

Brief Outline of Research Findings

Research in the past 6 months has again taken two directions: foundational investigations of case-based reasoning and the domain-specific investigation of case-based reasoning in labor mediation. A report of the foundational work has been submitted to the AAAI-87 Conference and is entitled "Capitalizing on Failure Through Case-Based Inference." The work reported in this paper addresses the issue of how previous failures can be used by a problem solver to help it avoid remaking mistakes.

Previous failures serve several purposes during problem solving. They provide warnings of the potential for failure in the current case, and they may also provide suggestions of what to do instead. Analyzing the potential for failure in a new case, a necessary part of capitalizing on an old failure, may require the problem solver to gather additional information, thus causing the problem solver to change its focus of attention. A previous failed case that was finally solved correctly can help the problem solver to change its point of view in interpreting a situation if that is what is necessary to avoid potential failure.

Errors in reasoning can happen during any problem solving step. The problem might have been misunderstood initially, resulting in incorrect classification of the problem or incorrect inferences during the problem elaboration phase. Since problem understanding is an early part of the problem solving cycle, such misunderstandings and incorrect inferences propagate through to the planning phase, resulting in a poor plan. A problem might be understood correctly and all the necessary details known about it, but might still be solved incorrectly because poor decisions were made while planning a solution. In general, such errors are due to faulty problem solving knowledge. The problem solver might not have complete knowledge, for example, about under what circumstances a particular planning policy or plan step is appropriate. Finally, a problem might be solved correctly but carried out incorrectly by the agent carrying out the plan, or unexpected circumstances might cause execution to fail. Reminding of a case where any of these things happened warns the the problem solver of the potential for the same type of error in the new case. If the previous case was finally resolved correctly, details of its correct resolution are used to provide suggestions for solving the new problem correctly.

Any time the problem solver encounters a case with a previous problem, it considers whether there is the potential for that problem in the new case. This may cause it to refocus itself until the potential for failure is determined, and if such potential is determined and the problem solver has to retract decisions made previous to the current one, then it must remake any decisions dependent on those decisions. Such processing, of course, requires that the problem solver be integrated with a reason-maintenance system that keeps track of the dependencies among its decisions. Other steps require that the reasoner record justifications for each of the decisions it makes. We have not done a great deal of work in these areas, but our experience so far leads us to believe that a standard truth maintenance system (Doyle, 1979, McAllester, 1980, DeKleer, 1986) is not adequate to do all of the work we need such a system to do. In particular, in addition to its standard bookkeeping functions, such a system will need strategies or policies to follow in making decisions about how to make the world consistent when a condition check fails, or will need to interact with a reasoner that can make such decisions. While it is standard for a truth maintenance system to retract decisions that are inconsistent and to propagate those retractions as far as it needs to, in the problem solving situation we are looking at, it is often more advantageous to try to satisfy constraints in a different way (e.g., to replace a retracted value with another that satisfies the necessary constraints).

Students are currently implementing some of these ideas, though not in the MEDIATOR per se, and not with ARO funds. Our JULIA program, which plans meals, uses the processes outlined above to avoid remaking mistakes. JULIA's representational structures are designed to store adequate knowledge about reasoning done in previous cases to allow it to do this. Also in JULIA, we are beginning to investigate the interactions between multiple problem solving processes, e.g., the case-based reasoner, the reason maintenance system, and a set of from-scratch problem solvers. In the MEDIATOR, case-based reasoning is always attempted before trying general purpose methods. This has worked well and allowed us to ignore questions of control, but we are not searching for more sophisticated answers to the problem of integrating case-based reasoning with other sorts of reasoning.

In her work on the PERSUADER, Katia Sycara continues to investigate the interactions between heuristic methods (in this case, case-based reasoning) and mathematical methods (her adaptation of utility theory for dealing with multiple competing goals). A paper she has recently had accepted to a book on techniques for operations research on this topic is attached. I expect that her Ph.D. thesis will in large part explain her conclusions about such interactions, and that, because she is equally comfortable in both the heuristic and mathematical worlds, that in her future work, it will be the place where

she will make her best contributions.

In the past 6 months, Katia has also spent considerable time determining what kinds of knowledge structures are best suited for holding the knowledge necessary for mediation, and has invented an adaptive knowledge structure called a *Situation Assessment Package* (or SAP) that, in the tradition of Schank's (1982) TOPs, store recognition criteria for different kinds of conflict situations and strategic knowledge for resolving each kind of conflict and also organize experiences using those strategies. In this way, novel uses of the strategies can be easily recalled and used when necessary. A paper she has submitted to the 1987 Cognitive Science Society Conference describing this work is attached. Katia is approximately 2 months from finishing her Ph.D. work, and I hope to send you copies of her Ph.D. thesis in the next semi-annual report. She will be continuing at Georgia Tech with funding under this grant until she begins her job (not yet selected) in mid-summer or fall, 1987.

PROGRESS REPORT

TWENTY COPIES REQUIRED

1. ARO PROPOSAL NUMBER: 21697-MA

2. PERIOD COVERED BY REPORT: 1 January 1987 - 30 June 1987

3. TITLE OF PROPOSAL: The Role of Experience in Common-Sense & Expert Problem Solving

4. CONTRACT OR GRANT NUMBER: DAAG29-85-K-0023

5. NAME OF INSTITUTION: Georgia Institute of Technology

6. AUTHORS OF REPORT: Janet L. Kolodner

7. LIST OF MANUSCRIPTS SUBMITTED OR PUBLISHED UNDER ARO SPONSORSHIP DURING THIS REPORTING PERIOD, INCLUDING JOURNAL REFERENCES:

Kolodner, Janet L., "Extending Problem Solver Performance Through Case-Based Inference", Proceedings of the Fourth International Machine Learning Workshop, June, 1987.

8. SCIENTIFIC PERSONNEL SUPPORTED BY THIS PROJECT AND DEGREES AWARDED DURING THIS REPORTING PERIOD:

Katia Sycara, awarded Ph.D. June, 1987.

Ph.D. Thesis entitled: Resolving Adversarial Conflicts: An Approach Integrating Case-Based and Analytic Methods

(preface attached, copies of thesis to be sent under separat e cover when available)

Janet L. Kolodner
Department of Computer Science
Georgia Institute of Technology
Atlanta, GA 30332

Brief Outline of Research Findings

Research in the past 6 months has again taken two directions: foundational investigations of case-based reasoning and the domain-specific investigation of case-based reasoning in labor mediation. A report of the foundational work has been presented at the Fourth International Machine Learning Workshop and is entitled "Extending Problem Solver Capabilities Through Case-Based Inference." The work reported in that paper brings together much of the work we have been doing in case-based reasoning over the past few years.

In particular, our current analysis of case-based reasoning states that there are two primary types of case-based analysis:* *comparison-based reasoning*, in which a solution from a previous case is borrowed and then modified based on differences between the current and previous cases; and *problem-reduction driven case-based reasoning*, in which parts of solutions of several previous cases are borrowed and then pieced together to make a complete solution to the new problem. Case-based inferences made during problem-reduction driven case-based reasoning can be made by either of these two methods. My analysis has also been of what can be borrowed from a previous case, and I have come up with three answers to that: the solution itself, the means of deriving that solution, or the conditions under which the previous solution was derived. I do not know yet under what circumstances each is appropriate. Nor have all details of these taxonomies been worked out.

Work in the labor mediation domain has been done by Katia Sycara. Most of her work in the past six months has been devoted to writing up her research in her Ph.D. thesis, and I am happy to report that she finished her dissertation in June. It is entitled "Resolving Adversarial Conflicts: An Approach Integrating Case-Based and Analytic Methods." Copies of her preface are attached to this report. Copies of her thesis will be sent under separate cover. Katia will begin work at the CMU Robotics Institute in September. She will be working in Mark Fox's group and will be attempting to apply what she knows about case-based reasoning and methodologies for conflict resolution to manufacturing and scheduling problems.

Katia's thesis provides the final report on the PERSUADER project. In that research, Katia has investigated the particular case-based reasoning method that I called comparison-based reasoning above, and that she calls prededent-based reasoning. Comparison-based (or precedent-based) reasoning is a method of deriving a solution to a new case by recalling one that is highly similar, computing the differences between the recalled and the new case, and based on those differences modifying or patching the old solution to fit the new situation. Case-based reasoning can also be used for this last step. Katia has shown in detail how comparison-based reasoning works for a particular domain, and just as importantly, shown under what circumstances it breaks down and what can be done when that happens. When no cases are available, her program employs analytic methods (in this case an adaptation of utility theory formulations) to mediate between goals and come up with a compromise solution. Any program that uses case-based reasoning will need some kind of "from-scratch" method when cases are not available, and one appropriate to the particular domain must be chosen.

Even when cases are available, comparison-based reasoning methods may not be appropriate. This is the case when the new case is different from what is expected in ways that predict that the usual types of solutions won't work. In labor/management disputes this happens when the company is being mismanaged, when the union or the company have goals that are out of line with the norms, and several other times. Katia's way of dealing with this type of situation is to classify it by its goal/plan interactions (much as Schank suggests in his formulation of TOPs), and to use knowledge about dealing with those abstract kinds of situations to solve the problem. For example, if the company is being mismanaged, one applies "mismanagement remedies" in coming up with a solution. One mismanagement remedy is to punish those who are doing the mismanagement by placing an overseer over them to make sure they will do things correctly in the future. In the labor/management domain, this might translate into placing union members on the board of directors. An interesting aside to this method is that while it is hard in general to specialize general strategies or remedies to specific new kinds of situations, once it has been done the case can be remembered and case-based reasoning can also help here.

Katia's thesis details the processes and knowledge structures necessary for each of these tasks.

---

* Term due to Gary Klein.

EXTENDING PROBLEM SOLVER CAPABILITIES THROUGH CASE-BASED INFERENCE

Janet L. Kolodner

Abstract

In case-based reasoning, the problem solver makes its inferences based directly on previous cases rather than by the more traditional approach of using general knowledge. Case-based reasoning results in several enhancements to problem solving behavior over time. First, recall of previous failures warns the problem solver of potential for failure and allows the problem solver to avoid making mistakes made previously. Second, previous decisions that have been made previously are suggested to the problem solver so that its decisions do not have to all be made from scratch. This lessens the search space and also is a way of short-cutting the constraint satisfaction process. Third, if abstract schemata can be derived from cases that have been seen previously, generalized knowledge can be augmented. This allows real shortcuts in problem solving. Decisions that previously took several reasoning steps to make may be possible through application of a generalized schema.

GIT-ICS-87/26

RESOLVING ADVERSARIAL CONFLICTS: AN APPROACH INTEGRATING CASE-BASED AND ANALYTIC METHODS

Ekaterini P. Sycara

School of Information and Computer Science
Georgia Institute of Technology
Atlanta, Georgia 30332

June, 1987

A thesis presented to the faculty of the Division of Graduate Studies in partial fulfillment for the degree of Doctor of Philosophy in the School of Information and Computer Science.

ACKNOWLEDGEMENTS

This research would not have been possible without the efforts and support of many people.

My greatest debt is to Janet Kolodner, my advisor and friend, who first interested me in AI and then focused my interest on case-based reasoning. Without Janet's unflagging interest, suggestions, and criticisms, this work could not have approached its full scope or attention to detail.

For knowledge of labor mediation I am indebted to Sherman Dallas. He has proved as patient and competent a domain expert as an AI researcher could hope to find.

Rich DeMillo deserves special thanks for agreeing to be on a thesis committee of mine for a second time. The comments and suggestions from my other committee members, Rich Cullingford and Larry Barsalou, proved invariably useful and to the point.

Without the day to day support and camaraderie of the students in the AI group this research would have seemed more drudgery and less fun. Bob Simpson, especially, was a friend, and his contributions both to my morale and the quality of this research can not go unmentioned.

This research was funded in part by the Army Research Office under contract No. DAAG 29-85-K-0023.

Finally, I wish to acknowledge my family who has lived with me through this ordeal by research providing a little help with a lot of grace.

By- Ekaterini P. Sycara

ABSTRACT

This thesis addresses the problem of finding compromise solutions to multi-agent conflicts. This is a difficult problem since the compromise choices that a problem solver has for continuum-valued goals are infinite, and the agents need to be persuaded to partially abandon goals during problem solving.

To deal with these difficulties, we propose an integration of:

-- Heuristic methods: Use of past cases similar to the current problem

-- Analytic methods: Application of multi-attribute utility theory to many decision makers

Past problem solving episodes similar to the current one are used to focus on the relevant parts of a problem, form a basis for analogical reasoning, avoid past mistakes, and suggest argumentation strategies. Utility theory is used to identify feasible compromises, evaluate whether a contemplated solution is an improvement on a previously rejected one and provide a computational formalism for persuasive argumentation.

We present the processes mentioned above and the knowledge sources that support them. Our examples are taken from the domain of labor mediation and are implemented in a computer program, called the PERSUADER. The PERSUADER functions as a mediator in hypothetical labor negotiations. Using the methods described, it suggests appropriate settlements to the disputants. If a suggested compromise is rejected, the PERSUADER attempts to either modify the settlement or the opposing party's "view" of the settlement using the same methods.

Katerina P. Sycara

PREFACE

This thesis presents a theory of problem solving required to resolve multi-agent conflicts. The problem solver acts as a mediator guiding the agents toward a solution acceptable to all.

The bulk of Artificial Intelligence work concentrates on problem solving situations with only one agent. The presence of more than one agent, however, is an unavoidable reality. People must plan their everyday activities taking into consideration the goals and plans of others, which might be in conflict or in concord with their own. Virtually all AI researchers who have looked at multi-agent interactions have assumed that common goals of multiple agents are non-conflicting (i.e., in concord) (Cammarata, 83; Davis, 83; Georgeff, 84). Their work has thus focused on how these agents can best help each other in achieving their common ends.

Our work, on the other hand, investigates situations where agents have conflicting goals, and where compromise solutions to the conflict would be beneficial for everyone. There are many real-world situations where this is necessary and, as technology advances, there will be increasing need to automate the process. In an automated factory, for example, robots might compete for limited resources. An automated battle manager may need to coordinate a schedule with another automated or human battle manager, while observing the priorities of its owner. A job shop scheduler (Fox, 1982) must schedule orders among available machines in such a way that his "clients" are all adequately satisfied. Even in situations where all agents are assumed to have a common goal, sub-goal conflicts may arise. For example, in a distributed problem solving environment, machines involved in solving a single problem may conflict over use of various computational resources. As research in distributed computing progresses, such situations will be very common.

Finding a compromise solution for multiple conflicting goals is difficult. The problem solver must somehow find a settlement that includes "suitable" values for each issue on which all agents will agree. For continuum-valued issues, the choices that such a problem solver faces are infinite. A blind trial and search process is obviously hopeless. Hierarchical decomposition of the problem to smaller problems each of which is easier to solve (a common method for AI problem solvers) cannot be employed since a compromise solution may be a "package" whose parts are strongly interconnected and interacting. These difficulties are compounded by the absence of a coherent set of constraints that could guide search through the space of all possible settlements.

In real life, conflict resolution is usually a lengthy and iterative process during which a problem solver (usually a mediator) comes up with a compromise solution that he proposes to the parties. If the solution is rejected, the problem solver must either modify the settlement or convince the dissenting party to modify his goals and accept the settlement.

Our model of conflict resolution parallels the real world model. In our model, there are three parts to the conflict resolution process: generation of an initial compromise, modification of a proposed settlement based on feedback from the dissenting party, and persuasive argumentation. As in real life, a compromise is first generated, then proposed, and feedback from the parties is obtained. A decision is made about whether to modify the proposed solution or persuade the disagreeing party. One or the other is done and the process cycles iteratively until resolution is reached.

We present three methods that can be used to implement these processes: case-based reasoning based on precedents (Kolodner et al., 1985), preference analysis (Sycara, 1987a), and situation assessment (Sycara, 1987b). All three methods are used to generate an initial compromise solution. The case-based reasoning method we use involves choosing an appropriate previous case, identifying differences between that case and the current situation, and then, using those differences to criticize and modify the previous solution to fit the current problem. Preference analysis is used to find compromise solutions when appropriate past cases are not available. It involves using degrees of utility that the parties attach to their goals to calculate the degree of satisfaction of each party with respect to a proposed settlement. Using this method, a reasoner can model the tradeoffs that an agent is willing to make among his goals

and predict which compromise the agent will be most willing to accept. When a problem cannot be solved by either of these methods because of some unusual aspects that it has, situation assessment is used. This process accesses domain independent knowledge structures that describe the causal structure of the situation and hold strategies for dealing with it. Either a generalized strategy is applied or case-based reasoning is used to specialize a general strategy to the new problem.

When a solution is rejected, a problem solver either generates arguments to convince the dissenting party to agree, or modifies the settlement to make it more acceptable. Several kinds of knowledge are used to generate persuasive arguments. When a previous appropriate argument is available, case-based reasoning is used to modify it to fit the current case. When a previous argument is not available, the problem solver constructs an argument by first analyzing the preferences of the agents to see which beliefs need to be modified and then generating an argument based on the goals of the agents. Modifying a rejected settlement is done by two methods: case-based reasoning and preference analysis. Case-based reasoning involves identifying a previous settlement that has been rejected for the same reason and adapting the resolution plan to perform appropriate modification to the currently rejected solution. Preference analysis involves using the payoff that the rejected solution gives the parties as a criterion of solution improvement.

The examples in this research are taken from the domain of labor mediation and are implemented in a computer program called the PERSUADER. The PERSUADER functions as a mediator in hypothetical labor negotiations. Given a labor management dispute, the program uses the methods described to suggest an appropriate settlement to the disputants. If a suggested compromise is rejected, the PERSUADER attempts to either modify the settlement by case-based reasoning and preference analysis or to modify the opposing party's "view" of the settlement through persuasive argumentation.

This research presents models of (a) resolution of multiple conflicting goals, (b) persuasive argumentation, (c) planning for partial goal satisfaction, and (d) integration of heuristic/case-based and analytic methods. As a model of conflict resolution, the PERSUADER suggests what the ingredients of resolution strategies must be. As a model for partial goal satisfaction, it has implications for human decision making. As a system that embodies a theory of persuasive argumentation, it presents a novel framework for the study of attitude and belief modification. It also demonstrates the usefulness of case-based reasoning in a variety of tasks necessary for problem solving in complex domains. The novelty of the research is not only that it addresses problems little studied before, but also that it addresses them in an integrated framework.

Another contribution of this research is in the methodology that it uses to support the theory. This work has proposed a methodology that integrates analytic (preference analysis) and heuristic (case-based reasoning and situation assessment) methods in problem solving. The integration of analytic and heuristic methods provides the following advantages:

1. The problem solver does not break down when heuristic methods fail.

2. Some stages of problem solving might be better suited to one method over the other. Providing both methods gives the problem solver the flexibility to treat each problem solving stage via its most natural solution method.

3. Heuristic methods alone do not allow fine tuning of a solution to a sufficiently small level of granularity. Rather, they are used to come up with a ballpark solution and analytic methods refine it.

There are several ways that the methods interact. The analytic method provides a way to construct a solution when heuristic methods are not capable. The heuristic methods support the analytic by providing needed information that it would be tedious to obtain otherwise. The analytic method provides a means to evaluate a solution constructed by modification of a heuristically derived solution.

Our theory of persuasive argumentation provides an explicit model of belief and behavior, and how they can be changed by arguments. This theory of attitude change based on the manipulation of goals presents a dynamic alternative to to social psychological theories based on balance (Festinger, 1957; Heider, 1958) and is suited to more complex situations. Both utility curves and goal graphs serve as representational vehicles for expression of beliefs and partial goal satisfaction and allow a problem solver to incorporate partial goal fulfillment strategies into his reasoning. In contrast to the above mentioned earlier theories that were limited to specification of attributes of arguments, the explicit representation of beliefs in our model allows it to specify the arguments themselves. In other words our theory is *constructive* rather than *prescriptive*.

The PERSUADER makes explicit what knowledge is needed in negotiation, how it is represented and organized, and how it is used to make decisions during negotiations. The PERSUADER provides a normative reference with which to compare and evaluate actual negotiations. By making the knowledge and the mechanisms explicit, the PERSUADER articulates testable hypotheses that might help in understanding negotiations.

This research breaks new ground in solving problems defined in psychological dimensions. Preference analysis, situation assessment and the goal-based theory of argumentation provide tools for solving human problems with subjective and irrational components. The ability to accomodate such "human" characteristics will become crucial as artificial intelligence seeks to move from the laboratory to general use.

GIT-ICS-85/19

ARGUMENTS OF PERSUASION: TWO PAPERS

Katia Sycara-Cyranski

June, 1985

School of Information and Computer Science
Georgia Institute of Technology
Atlanta, Georgia 30332

This research has been supported in part by NSF Grant No. IST-8317711 and in part by ARO Grant No. DAAG 29-85-K-0023. I would like to thank Janet Kolodner and Bob Simpson for helpful discussions and comments.

FOREWORD

The theme that is common to the two papers in the present report is persuasive argumentation in the context of mediation of labor management disputes. Collective bargaining is the process through which a union and a company reach agreement over a contract. If, during the course of negoriations an impasse occurs, a mediator is called in to help the parties resolve it. A mediator's role has basically two components: (a) to help the parties explore feasible alternative settlements and (b) to persuade the parties to reach an agreement. In this report, we present results with respect to the second task. The implementation of our ideas is embodied in a computer program, the PERSUADER.

The first of the papers, *Persuasive Argumentation in Resolution of Collective Bargaining Impasses*, explores the situation where the mediator, faced with a disputant who objects to a setllement, recalls arguments that she has used in similar situations in the past. If no appropriate arguments can be recalled, the mediator has to construct them. This case is explored in the second paper, *Arguments of Persuasion in Labour Mediation*. The order of presentation of the two papers seems more logical, although the paper that is presented first was written later. Thus, some of our ideas are more evolved in the first paper.

PERSUASIVE ARGUMENTATION IN RESOLUTION OF COLLECTIVE BARGAINING IMPASSES

Katia Sycara-Cyranski

March, 1985

School of Information and Computer Science
Georgia Institute of Technology
Atlanta, Georgia 30332

ABSTRACT

In this paper we present a process model that uses past experience in generating arguments of persuasion. We view persuasive argumentation as an instance of problem solving. As such, we employ knowledge organization ideas and problem solving techniques that have been advocated in an analogical view of problem solving. To illustrate our ideas, we use the domain of mediation of labor disputes. Our model is embodied in the PERSUADER, a computer program that gives advice in collective bargaining mediation.

This paper will appear in the Proceedings of the Seventh Annual Conference of The Cognitive Science Society, August, 1985.

1. *Introduction*

Persuasion has been and will continue to be a chief instrument in the conduct of human affairs. Arguments are the means by which persuasion is effected. During persuasive argumentation, an agent, the persuader attempts to change the beliefs of another agent, the persuadee. In this paper, we present a process model of persuasive argumentation that uses past experience to create new arguments. Our model is influenced by the work of Kolodner and Simpson (1984) on case based reasoning in problem solving. We use the domain of labor management disputes to illustrate our points.

Traditionally, the psychological literature has treated persuasion as a process of communication (Bettinghaus, 1968, Brembeck and Howell, 1976). In our model, persuasive argumentation is viewed as an instance of *problem solving*. The goal of the persuader as problem solver is to convince the persuadee to accept a particular proposition.* In labor mediation, the mediator is the persuader and the union or company the persuadee. When an impasse is reached in contract negotiations, a mediator is usually called in. The goal of the mediator is to convince the parties to reach a mutually acceptable contract without a strike.

This goal is achieved incrementally through many rounds of persuasive argumentation. In each round, the mediator tries to narrow the disputants' differences with respect to a contract issue, by convincing them to move towards a common value. In such cases, mediators traditionally use well-known persuasive arguments. An example of such an argument is that the adoption of seniority reduces labor turnover. These arguments and the appropriate ways to use them are identified in books on collective bargaining (Herman and Kuhn, 1981, Randle, 1951).

We view these arguments as plans that the mediator uses to achieve the goal of changing a party's position with respect to a contract issue. For a plan to be applicable, its preconditions have to be satisfied. The main factor determining the effectiveness of arguments of persuasion is the attitudes and beliefs of the persuadee (Abelson, 1959). The persuader has such a model of the persuadee in mind, to which he is addressing the persuasive arguments. We consider the persuadee model as part of the argument plan's preconditions. Another part of the preconditions is the economic context of the dispute. Argument plans are known by the mediator and are instantiated when the present case matches their preconditions. The task of the persuader is to decide the applicability of these plans to the situation at hand. To motivate our exposition, we present the following example:

---

* In adversarial argumentation the arguer does not attempt to change the beliefs of the interlocutor (Flowers, et. al., 1982).

The Yellow-Jackets textile company involved in a collective bargaining case refuses to grant the workers plantwide seniority for promotions and layoffs. The mediator suggests that seniority improves worker morale, resulting in more efficient plant operation and, consequently, decrease of production cost. The company points out that quite a number of key employees are junior and, during a layoff, they would be the first to go. This would impede the operation of the plant. The mediator, having this additional information, recalls a similar situation where the following solution was found: an exception in seniority for a number of key employees was accepted by the union in exchange for super-seniority for union officers and stewards. The mediator proposes this compromise to the company, which agrees.

In this example, the mediator proposes an argument plan that she thinks is suitable to the particular situation. To generate the initial argument, the mediator recalls relevant economic factors, important goals of similar persuadees, and experiences with the same contract issue. Since these three forms of information might come from different mediation experiences, the mediator needs to combine information from the individual available schemata, constructing the most appropriate combination for the present situation. We call this schema the argumentation precedent. In this case the precedent includes the information that efficient plant operation is an important company goal, that seniority improves worker morale leading to worker efficiency, that the economic conditions are recession, and that the majority of textile industry contracts have seniority provisions. The precedent is used as a set of preconditions, against which arguments are tested for applicability.

The next two figures show the conceptual content of the initial argument plan and argumentation precedent for the above example. Space limitations prohibit a full explanation.

THE PERSUADER'S INITIAL ARGUMENT PLAN

```
persuadee: Yellow-Jackets company
issue    : *seniority*
preconditions: argumentation precedent (below)
claim: Increased plant efficiency comes from granting seniority
persuader-goal: Change weight of issue  ;see section on strategies
argument-type: Self-interest            ;see section on convincing power
strength: .7                            ;see section on convincing power
```

Figure 0-1

ARGUMENTATION PRECEDENT

persuadee-model: goals of the Yelow-Jackets, including their relative importance
economic-context: recession, unemployment in the textile industry,...

Figure 0-2

2. *The overall model*

We present the overall process model for persuasive argumetation in Figure 3.

PROCESS MODEL OF PERSUASIVE ARGUMENTATION

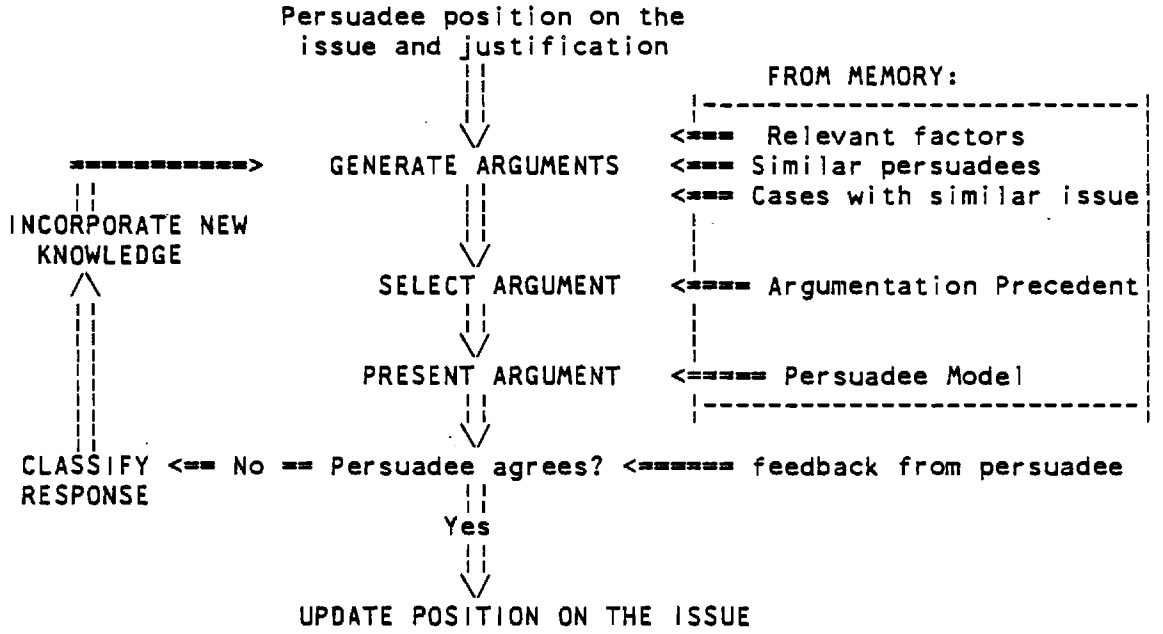


Figure 0-3

The input to the argumentation process is the persuadee's position on an issue and the position he needs to be convinced of. In mediation, these correspond to the value of the contract issue that the party has rejected and the mediator's proposal. The first stage in

persuasive argumentation is to generate potentially applicable arguments using the contract issue as a probe. The most appropriate argument is then selected from the retrieved ones. This is done by using the argumentation precedent, as a set of preconditions against which the potential effectiveness of retrieved arguments is tested. Consider, for example, the argument that *the adoption of seniority for promotions reduces grievances*. The rationale is that seniority is a criterion well-understood by the workers and thus will eliminate potential complaints of unfairness. The strength of this argument for the company depends directly on the importance of reducing grievances as a company goal. Relative importance of goals is included in the argumentation precedent.

Next, the persuader presents the selected argument. If the persuadee agrees, the appropriate update of the settlement is made, namely that there is agreement on this issue. If the persuadee disagrees, the reasons for the disagreement are analyzed for new information that could alter subsequent argumentation, such as new information about the persuadee's concerns (e.g., the company's concern about key employees), new information about economic factors (e.g., the strength of foreign competition), and corrected inferences about the relative importance of the persuadee's goals. The mediator incorporates the new knowledge into the argumentation precedent. In this way, the argument preconditions are dynamically learned as a result of comparing successful and failed applications of the argument. The process of generating potentially applicable arguments is then repeated, testing argument effectiveness against the updated argumentation precedent. A new, more convincing argument is selected for presentation.

3. The persuadee model

The persuadee model, used during argument generation, selection, and presentation, contains the attitudes and beliefs of the persuadee. These are represented in terms of his collection of goals and their relative importance. Goals of a union or company negotiator are of two types: personal career goals and the goals of the union or company he represents. We represent these goals in goal trees (Carbonell, 1979). In the subsequent figure we depict the partial goal tree of a company.

COMPANY GOAL-TREE

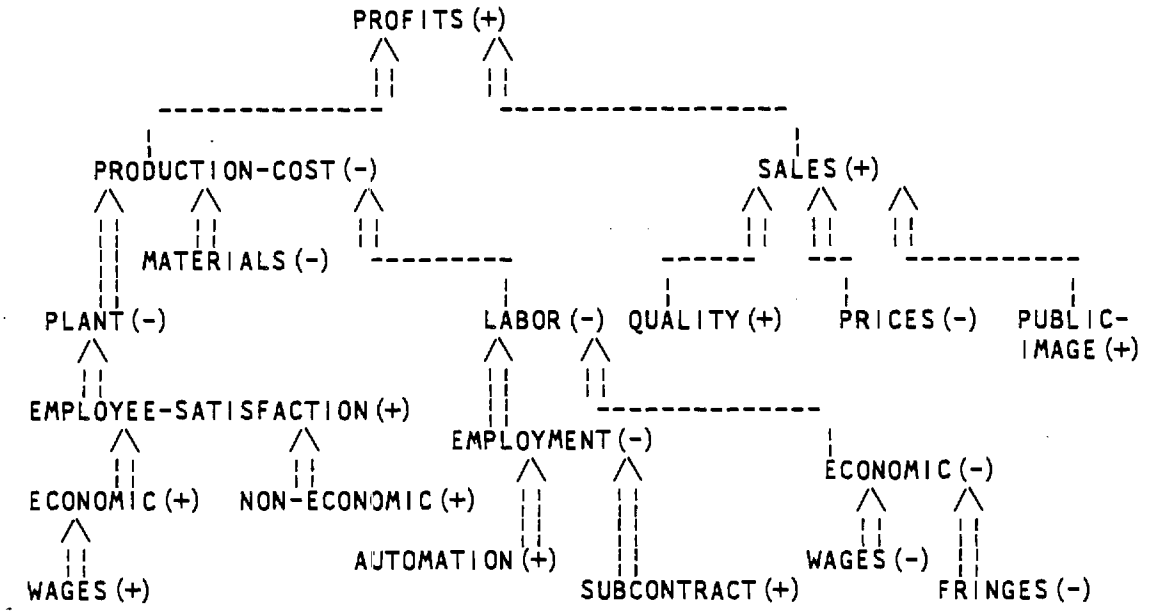


Figure 0-4

The notation for the relationships among goals in the tree is adopted from Spohrer and Riesbeck, (1984). A (+) sign corresponds to the goal of increasing the particular quantity to which it refers, while a (-) sign corresponds to decreasing the quantity. For example, increasing profits, PROFITS(+), which occupies the root of the goal tree, represents the company's highest level goal. The children of a node, connected to it through *support* links, denote the subgoals through which the supergoal is satisfied. For example, profits can be raised, PROFITS(+), by decreasing production costs, PRODUCTION-COST(-), or by increasing sales, SALES(+).

Also included in a goal tree is the relative importance of the party's goals, though for simplicity, this is not shown in the figure. The figure depicts a "prototype" instance (Rosch, 1977) of a company's goals. Goal trees vary with particular negotiators and companies (unions), and the best one possible is needed to construct effective arguments. When a persuader is faced with an unknown persuadee, he can use a prototype goal tree, or a persuadee model by transferring characteristics from the goal tree of a previously encountered and similar persuadee.

4. Effective persuasion

There are two central issues in selecting the most effective argument plan: first, the persuader's goal, namely in what way does he want to change the persuadee's beliefs; and second, how to do it most convincingly. The first issue involves strategies of persuasion and the second, criteria for the persuasive power of arguments.

4.1 Strategies for argument plan selection

One measure of successful persuasion is the acceptance of the proposed solution by the parties. In mediation, this means the willingness of a party to accept a suggestion regarding a particular contract issue. This willingness depends on the party's assessment of the monetary value of that issue and the issue's importance. Hence, a mediator has two possible goals in convincing a party to accept a previously rejected issue:

1) changing the importance that the party attaches to the issue, or
2) changing the party's assessment of the issue's proposed monetary value

The argumentation strategies used to accomplish these goals determine how the argument plan selection is done. For example, if the persuader's goal is to change the importance accorded an issue by the persuadee, and he chooses to use the first strategy, then a threatening argument plan has to be used.

Three argumentation strategies can be used to accomplish the first goal:

(a) indicate possible unpleasant consequences of the present demand
(b) propose alternatives
(c) produce evidence showing that the particular proposal promotes an important goal of the persuadee

To illustrate the first strategy, suppose a union rejects a wage settlement. The mediator tells the union that if the company is forced to grant higher wages, it will become non-competitive and therefore will be forced to lay off workers. If an important union goal is preservation of employment for its members, then the union will abandon its goal of higher wages in order to satisfy its employment goal.

Two strategies can be used to accomplish the second goal:

(d) recall a "counterexample" from the persuadee's record of contracts
(e) recall examples of similar unions (companies) having settled for the proposed value or less (more)

To illustrate the last strategy, consider a union's rejection of an increase of 10 cents per worker per hour in health benefits as unacceptably low. The mediator presents contracts signed by the same or other unions which incorporate an equal or lower increase. This arg-

ument is effective because perception of "low" or "high" values is determined by *prevailing practice*, namely what settlements similar disputants have agreed to.

4.2 The convincing power of arguments

For persuasion to be effective, the appropriate type of argument has to be presented in each situation. Examining a great number of arguments used in labor mediation, we have identified six categories of argument plan types. They have general applicability, although we will use examples from the mediation domain to clarify their use. We present them in a default ordering of persuasive power (from weakest to strongest):

1) Appeal to universal principle

In using a universal principle, the persuader appeals to some core belief of the persuadee as support for the argument. An example is the argument that a particular wage value does not afford the workers a "decent living standard". Arguments of this type are generally weak, since they appeal to moral principles rather than to the economic realities. However, if "public image" is an important company goal, arguments of this type take on added power.

2) Appeal to "minor standard"

"Minor standards" provide exceptions as a basis for refutation of arguments based on prevailing practice. In mediation, "minor standards" are used as justifications to propose settlements to the employees of one company that differ from settlements within the industry in general. Examples of minor standards include steadiness of employment and hazardous work (Elkouri and Elkouri, 1973).

3) Appeal to "prevailing practice" standard

People's attitudes and goals are strongly influenced by the groups to which they belong. They use the achievements of their peers as a standard with which to compare their situation and expectations. In mediation, this corresponds to the *prevailing practice standard*. Prevailing practice is the most frequently used argument in labor mediation. Its credibility derives from economic reality. A company cannot underpay its employees for fear of loosing them to competitors; a union cannot insist on concessions much above what is given in the industry, for fear of lay-offs.

4) Appeal to precedents as counterexamples

Use of precedents as counterexamples provides a strategy to convince a persuadee that his claim is not as tenable as he would like to think. The power of counterexamples lies in their ability to point out contradictions between the claimed and the actual behavior of the persuadee. Psychological consistency theories (Heider, 1958, Festinger, 1957, Osgood and Tannenbaum, 1955) give evidence for the persuasive power of counterexamples.

5) Appeal to self-interest

The persuasive power of these arguments depends on the importance of the goal that is claimed to be promoted by the adoption of the persuader's proposal. People will substitute the satisfaction of a lesser goal for a more important one. An example of such an argument is the

acceptance by a company of seniority, because it reduces labor turnover, despite the resulting curtailment in management rights.

6) Threats

People want to satisfy their goals, so threatening an important goal of a persuadee is the most effective of arguments. In labor management disputes, the threat of a strike is the most frequently used and clearly the most powerful argument. However, there are other threats that can be very persuasive, as when a food-processing company's employees threaten to "leak" news of health violations at the plant. The mediator's role here is to convince the company that the employees will carry out their threat and that similar tactics have damaged recalcitrant companies in the past.

5. Summary and future work

We have presented a portion of the reasoning and domain knowledge necessary in a process model of persuasive argumentation, and given examples from the domain of labor mediation. In this paper, we have concentrated mainly on the task of argument selection. Important factors in this selection are the persuadee model, the argumentation strategies and the convincing power of arguments. Many issues have not been addressed. For example, what is the exact algorithm to construct the argumentation precedent, what is the role of feedback, what is the most appropriate memory organization.

6. References

Abelson, H. (1959). Persuasion.... New York, NY: Springer Publishing Company, Inc.

Bettinghaus, E. (1968). Persuasive Argumentation. New York, NY: Holt, Reinhart and Winston.

Brembeck, W. and Howell, W. (1976). Persuasion: A means of social influence. Englewood Cliffs, NJ: Prentice-Hall, Inc.

Carbonell, J. G. (1979). Subjective Understanding: Computer Models of Belief Systems. Doctoral dissertation. Yale University Research Report #150.

Elkouri, F. and Elkouri, E. (1973). How Arbitration Works. Washington, DC: The Bureau of National Affairs, Inc.

Flowers, M., McGuire, R. and Birnbaum, L. (1982). Adversary arguments and the logic of personal attacks. In W. G. Lehnert and M. H. Ringle (Eds). Strategies for Natural Language Processing, Hillsdale, NJ: Lawrence Erlbaum Associates.

Festinger, L. (1957). A theory of cognitive dissonance. Stanford, Calif: Stanford University Press.

Heider, F. (1958). The Psychology of Interpersonal Relations. New York, NY: Wiley.

Herman, E. and Kuhn, A. (1981). Collective Bargaining and Labor Relations. Englewood Cliffs, NJ: Prentice-Hall, Inc.

Kolodner, J. and Simpson, R. (1984). Experience and Problem-Solving: A Framework. In Proceedings of the Sixth Annual Conference of the Cognitive Science Society. June 28-30, Boulder, Colorado.

Osgood, C. and Tannenbaum, P. (1955). The principle of congruity in the Prediction of Attitude Change. In Psychological Review 62, pp. 42-55.

Randle, W. (1951). Collective Bargaining: Principles and Practices. Cambridge, Mass.: The Riverside Press.

Rosch, E. (1977). Classification of real-world objects: origins and representations in cognition. In Johnson-Laird, P. N. and Wason, P. C. (Eds.) Thinking, Readings in Cognitive Science. Cambridge: Cambridge University Press.

Spohrer, J. C. and Riesbeck, K. (1984). Reasoning-driven Memory Modification in the Economics Domain. Yale University Research Report #308.

ARGUMENTS OF PERSUASION IN LABOUR MEDIATION

Katia Sycara-Cyranski

January, 1985

School of Information and Computer Science
Georgia Institute of Technology
Atlanta, Georgia 30332

ABSTRACT

In this paper we present a process model of the reasoning underlying arguments of persuasion and its embodiment in a computer program, the PERSUADER, which gives counsel for the resolution of impasses in collective bargaining. We show how goal trees can be searched to produce arguments involving economic quantities.

This paper will appear in the Proceedings of the Ninth International Joint Conference on Artificial Intelligence, August, 1985.

1. *Argumentation strategies*

Arguments of persuasion are those used by the participants in cooperative problem solving. "Convincing" someone to accept a proposition can be effected by two strategies: 1) showing that the proposition furthers the person's goals or 2) indicating how refusing the proposition threatens his goals. In labour negotiations, the second strategy is crucial. In this paper, we present a procedure for constructing threatening arguments during labour mediation to resolve impasses in collective bargaining.

Collective bargaining is the process through which a company and a union arrive at a contract. Argumentation is used to persuade the opposing party to grant concessions, to support one's own demands, and to thwart attempts by the opposition to gain concessions from one's own side. A mediator, called in to help the two sides reach an agreement, engages in negotiations with each party in order to convince each to accept the necessary concessions. The final agreement constitutes a "package" which incorporates the tradeoffs that each party found acceptable. For example, while the union might not be completely satisfied with the wage increase that it achieved, it acquieses because the other clauses of the contract compensate for the wage sacrifice. By the time the mediator appears, most secondary issues have been settled. The mediator's job is to convince the parties to accept compromises on the last important issues.

Events 1 and 2 below illustrate how a mediator uses threatening arguments to accomplish this.

> EVENT 1. The company refuses to accept a particular wage settlement. The mediator argues that inefficient plant operation will occur from the resulting employee dissatisfaction.
>
> EVENT 2. The union refuses to accept a wage settlement. The mediator argues that if the company is forced to grant higher wages, it will become noncompetitive and therefore will be forced to lay off workers.

2. *Representing the parties' goals*

To generate an appropriate argument, the arguer must know the goals of the parties in question. We represent these goals in goal trees. Searching goal trees in order to understand and/or predict the behaviour of various actors has been investigated by Carbonell, (1979), Spohrer and Riesbeck, (1984), and Wilensky, (1983). In the subsequent two figures, we depict partial goal trees of a union and a company.

UNION GOAL-TREE

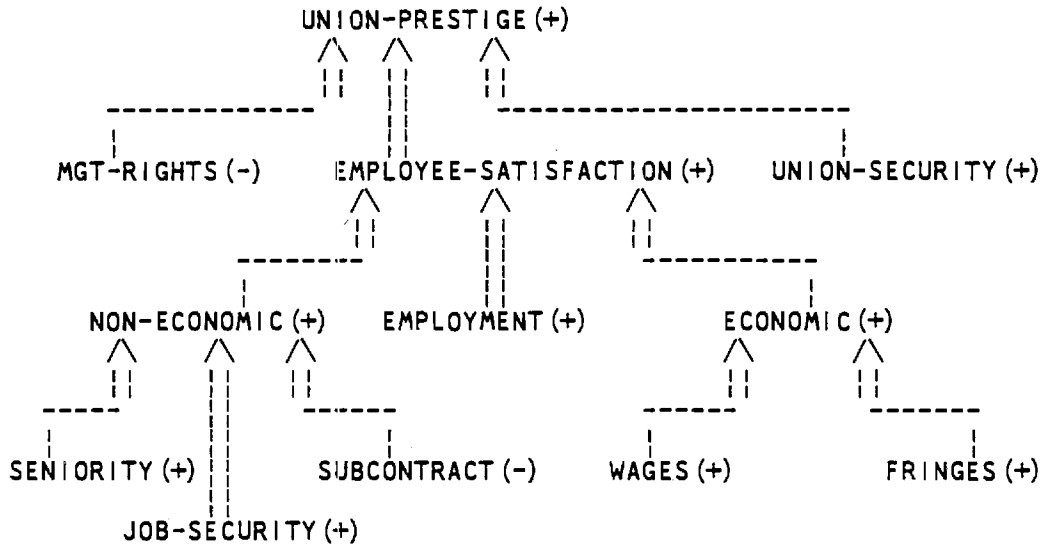


Figure 0-1

COMPANY GOAL-TREE

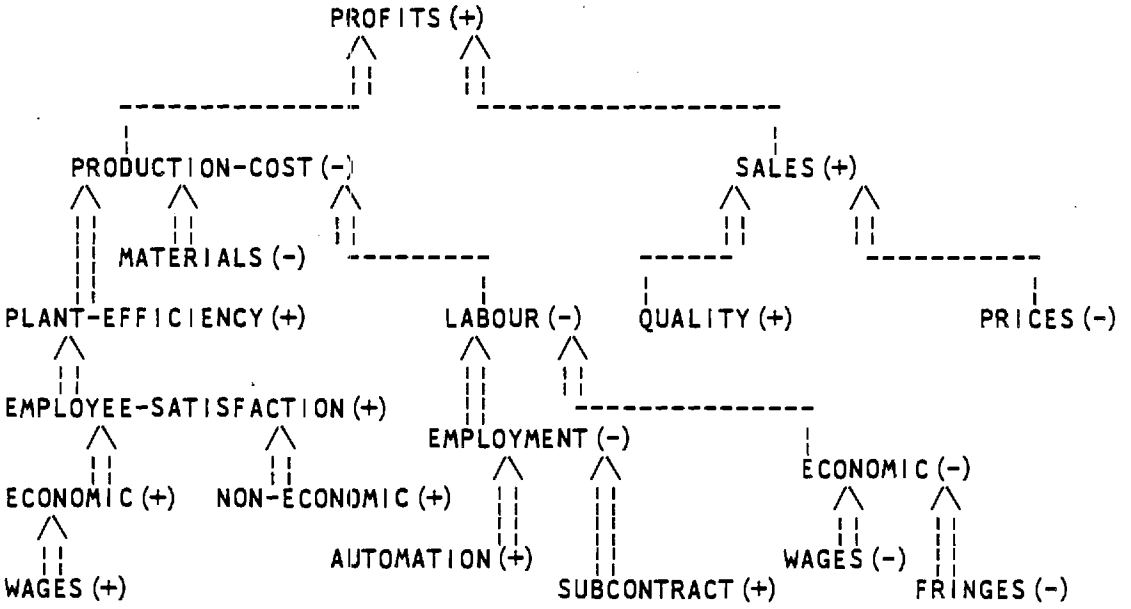


Figure 0-2

The relationships among goals are adapted from Spohrer and Riesbeck, (1984). A (+) sign corresponds to the goal of increasing the particular quantity to which it refers while a (-) sign corresponds to decreasing the quantity. For example, PROFITS(+) means that the company's highest level goal is to increase profits. A goal is violated by an action when the action opposes its sign. For example, if the company lays off employees, a reduction in employment, EMPLOYMENT(-), occurs, violating the union's goal EMPLOYMENT(+).

The children of a node, connected to it through support links, denote the subgoals through which the supergoal is satisfied. For example, in the company's goal tree, diminished labour costs can be achieved either by decreasing the economic concessions granted to the union, ECONOMIC(-), or by decreasing the number of employees, EMPLOYMENT(-). Thus, a path X to Y in a goal tree constitutes a causal chain that produces an explanation of the change in Y in terms of the change in X, assuming no other change has occured in the rest of the tree. The path WAGES(-) to PRODUCTION-COST(-) in the company's tree can be interpreted as: "Other things being equal, diminishing the cost of wages results in decreasing the cost of the economic concessions, which causes a decrease in labour costs, leading to a decrease in production costs".

A conflicting goal has a (+) sign in one goal tree and a (-) sign in the other. For example, the union's goal of increased employment for its members, (EMPLOYMENT(+) in the union's tree), conflicts with the company's goal of laying off employees, (EMPLOYMENT(-) in the company's tree). When, in one party's goal tree, the same goal exists in more than one place with opposite signs, an internal conflict exists for this party. Notice, for example, the internal conflict on economic concessions in the company's goal tree: The company wants to grant more economic concessions, ECONOMIC(+), in order to increase efficient plant operation, while simoultaneously it wants to decrease economic concessions, ECONOMIC(-), in order to reduce its labour costs.

The above representation, while allowing the arguer to do some qualitative reasoning (de Kleer and Brown, 1982) regarding the parties' goals, is clearly a crude approximation of reality. Not only the direction, but also the amount by which a quantity is being changed, is important for determining the acceptability of a proposed settlement. To simplify our explanation, we will assume that a mediator has a means of generating a reasonable value for each demand, and that her task is to generate convincing arguments for their acceptance.

3. Generating threatening arguments

Argument generation is guided by the goals of the parties. In addition, the processing depends on which party must be convinced. To convince the union, the strategy is to discover a company action which threatens one of the union's important goals. To convince the company, the strategy is to discover whether the company's refusal will result in a violation of an important company goal. Since the company controls the hirings, firings and concessions, both of these strategies require a goal directed search of the company's goal tree. The union goal tree is used to find the threatened union goals and their importance. The process assumes that the other party has agreed to the

proposed settlement.

Creating an argument to convince the union regarding issue X and change of quantity (*), (where (*) is either (+) or (-)), is as follows:

(1) Find out which company goals are violated by the union's refusal.

This is done by following the support links starting with X(NOT*) in the company's goal tree i.e., tracing the consequences for the company of the negation of its goal. The effects of negating X are propagated by changing the signs of X's ancestor goals along the path.

(2) Find out what compensating actions the company might carry out to offset the effects of negating its goal X.

This is done by considering the children Z1,...Zn of each goal Y found in step 1. To qualify as a threatening argument, a potential compensating action Zi has to satisfy three conditions: 1) it must be controllable by the company, 2) it must violate a union goal and 3) the importance for the union of this violated goal must be greater than the importance of the demand under discussion. If the third condition is not satisfied by Zi, its children are checked to see whether they satisfy conditions 1) to 3); otherwise, the subtree of Zi is pruned, and the siblings of Zi are considered in the same way. If some Zk proves suitable, a potential argument is saved. Whether or not an argument has been generated, steps 1 and 2 are repeated starting from Y. Thus, the whole set of arguments is generated.

Consider, for example, the generation of the argument used in Event 2. At issue was a decrease in wages. The process starts by following WAGES(+), a negation of company's goal WAGES(-) up the tree. Figure 3 shows the fragment of the company tree after propagation of WAGES(+) has started.

COMPANY TREE SEARCH

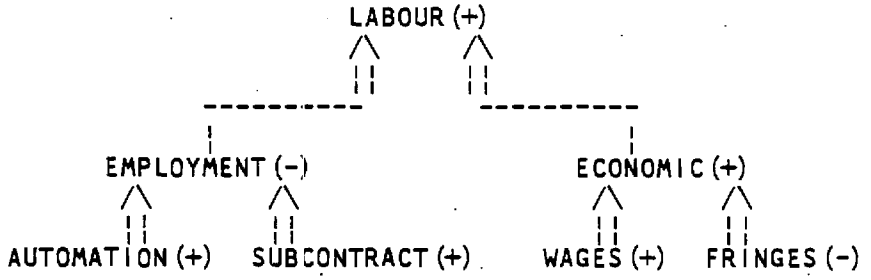


Figure 0-3

The search corresponds to a human mediator's reasoning: "Suppose the company increased the wages. Does this lead to violation of any union goal?" WAGES(+) leads to ECONOMIC(+). FRINGES(-) is considered as a possible action of the company to offset the increase in economic concessions. Thus, a possible argument might be: "If the company is forced to grant higher wages, it will reduce the granted fringes". Generating this argument depends on whether the company can reduce the fringes. Assuming that the fringes were not under negotiation in this case, the argument is rejected and the search continues from ECONOMIC(+). LABOUR(+) is reached, whose child, EMPLOYMENT(-) is controllable by the company and conflicts with the union goal EMPLOYMENT(+). Assuming EMPLOYMENT(+) is more important for the union than a wage increase, the argument "If the company is forced to grant higher wages, then it will lay off workers" is generated.

Generating an argument to convince the company about issue X is similar: the X(NOT*) path is followed in the company's goal tree. The mediator points out to the company the deleterious results that X(NOT*) has on one of its higher level goals.

4. An example from the PERSUADER

The PERSUADER is a program that generates appropriate contract proposals and tries to persuade the parties involved in the negotiation to accept them. In this example, it is handling an impasse in negotiations between a transit company and its union. The PERSUADER has generated a fair wage value, which the company has accepted and the union refused. The goals are organised as in figures 1 and 2. Importance of goals is expressed on a 0 to 10 scale. Here we see the PERSUADER trying to generate a threatening argument for the union.

```
---------------------------------------------------------------------
 Importance of #<M-WAGE-GOAL 22416471> is 6 for #<M-LOCAL 22405743>
 Searching #<M-TRANSIT-COMPANY 22412106> goal tree...
 Matching #<M-WAGE-GOAL 22416471> ...
 Set direction of #<M-WAGE-GOAL 22416471> to INCREASE...
 a INCREASE in #<M-WAGE-GOAL 22416471> by #<M-TRANSIT-COMPANY 22412106>
 will result in a INCREASE in #<M-ECONOMIC-GOAL 224224741>
---------------------------------------------------------------------
```

At this point, the PERSUADER considers fringe benefits but rejects it because it is not involved in the negotiation. It continues its search from #<M-ECONOMIC-GOAL 224224741>.

```
---------------------------------------------------------------------
 a INCREASE in #<M-ECONOMIC-GOAL 224224741>
 will result in a INCREASE in #<M-LABOUR-COST 22420554>
 To compensate, #<M-TRANSIT-COMPANY 22412106>
 will DECREASE #<M-EMPLOYMENT 22420562>
 which is contrary to #<M-LOCAL 22405743> goal
 Importance of #<M-EMPLOYMENT 22420562> is 8 for #<M-LOCAL 22405743>
 One possible argument found
---------------------------------------------------------------------
```

5. The convincing power of arguments

When the argument-generating process described above produces more than one potential argument, the best one must be chosen. One strategy is to try the "weakest" argument first, presenting "strong" arguments only if the weaker ones fail. This requires a means of ranking arguments according to their "convincing" power. The ranking follows the order of importance of the goals that the arguments threaten. In particular, the importance of the goals of a company (union) depends on the financial situation of the company, the state of the industry, the labour supply and the general economic climate. For example, the goal of reducing labour cost is more important for a company in an industry with high labour cost; if there is abundant labour supply in an area, the goal of employment is stronger for a union in that area. In this case, a threat to the union of layoffs has the greatest convincing power. Without enough information, the default ranking of arguments, from weakest to strongest, is:

1) Appeal to universal principle

Here, the arguer appeals to some moral belief of the interlocutor. For example, a particular wage value may not afford the workers a "decent living standard".

2) Appeal to precedents as counterexamples

Counterexamples point out contradictions between the claimed and the actual behaviour, thus threatening the credibility goal of a party.

3) Appeal to "prevailing practice" standard

Arguments based on this standard address economic goals. For example, a company cannot underpay its employees for fear of losing them to competitors; a union cannot insist on concessions much above what is given in the industry for fear of lay-offs.

4) Threat of Strike (Lockout)

A strike threatens to stop production, necessary for company profits. A lockout threatens the existence of the union. The mediator's role here is to convey to the recalcitrant party the dire consequences of the action.

6. Summary and related work

To generate a threatening argument to convince a union, the company's goal tree is searched to find company actions that will offset the effects of a union demand. To convince a company, the deleterious effects on the company of its demand are found by searching its goal tree. Though we have not addressed it, generation of arguments based on furthering of goals, can be done by a similar search.

While others have worked on argumentation, none so far has worked on persuasive arguments. The work of Flowers, et al. (1982) was concerned with adversary arguments and Spohrer and Riesbeck (1984) have investigated understanding causal relationships among economic quantities based on arguments given in newspaper articles.

7. References

Carbonell, J. G. (1979). Subjective Understanding: Computer Models of Belief Systems. Doctoral dissertation. Yale University Research Report #150.

de Kleer, J. and Brown, J.S. (1982). Foundations of Envisioning. In Proceedings of the AAAI-82. American Association of Artificial Intelligence, 1982.

Flowers, M., McGuire, R. and Birnbaum, L. (1982). Adversary arguments and the logic of personal attacks. In W. G. Lehnert and M. H. Ringle (Eds). Strategies for Natural Language Processing , Hillsdale, NJ: Lawrence Erlbaum Associates.

Spohrer, J. C. and Riesbeck, K. (1984). Reasoning-driven Memory Modification in the Economics Domain. Yale University Research Report #308.

Wilensky, R. (1983). Planning and Understanding: A Computational Approach to Human Reasoning. Reading, MA: Addison-Wesley Publishing Company.

GIT-ICS-85/18

A COMPUTER MODEL OF CASE-BASED REASONING IN PROBLEM SOLVING:

An Investigation in the Domain of Dispute Mediation

Robert L. Simpson, Jr.

June, 1985

School of Information and Computer Science
Georgia Institute of Technology
Atlanta, Georgia 30332

This research has been supported in part by the Air Force Institute of Technology, in part by NSF grants #IST-8116892 and #IST-8317711, and in part by the Army Research Office through grant #DAAG 29-85-K-0023.

ABSTRACT

Rather than approach each problem as a unique event, people often try to solve problems by recalling similar previous experiences as guides to problem solving. This analogical process, which we call case-based reasoning, seems to provide an explanation for the change in problem solving behavior of people over time. This research presents a computer process model of problem solving based on the use of case-based reasoning. The necessary reasoning processes, operational measures of similarity, and memory structures needed for effective storage and retrieval are presented via the specifications for an advisory system called the MEDIATOR, which offers advice on resolving common sense disputes. In this context, issues associated with enabling machines to dynamically adapt their reasoning and automatically recover from failure are discussed. The model of case-based problem solving which has been developed seems to offer promise as an integrated solution for some issues in problem solving, analogical reasoning, and machine learning.

ACKNOWLEDGEMENTS

"We give advice, but we cannot give the wisdom to profit by it." -- La Rochefoucauld

Numerous people and organizations have invested precious resources in helping to make this dissertation possible. When viewing my efforts from an investments perspective, I realize just how fortunate I am. For example, my employer, the United States Air Force through the auspices of the Air Force Institute of Technology, had the faith to invest the time (over four years) and money (my salary and tuition) to make my return to school possible. I'm indebted (literally) to them for their investment. Thanks (I think)! Besides the USAF, I have benefited indirectly from two National Science Foundation Grants: Nos. IST-8116892 and IST-8317711. My thanks to the NSF. In addition, my current supervisor, Bob Kahn, has been extremely understanding and supportive these last hectic weeks while I've finished the writing. Thanks Bob for your investment.

Laura, Bobby, and Karen have all made great personal sacrifices to support me in many ways. A wife and family have the right to expect that their spouse and father be present to help solve the numerous problems that are a part of rearing a family. Laura subordinated her personal and professional aspirations to insure that all her "school kids" were cared for properly. She was somewhat accustomed to the personal investment expected of a "military wife" in furthering a military man's career, but I doubt that she realized how much more of a personal investment was required to help see me through these last four years. All I can offer is to repeat the pledge I made to her in 1966 - to love, honor and cherish her so long as we both shall live. Bobby and Karen deserved more than a part-time father, theirs is an investment that I hope can be repaid in quality time if not in quantity time. You all taught me an important lesson in the strength derivable from positive family support. I love you so.

Since I've become a parent, I now appreciate how much of a personal investment is required of mothers and fathers on behalf of their offspring. I can't begin to express my gratitude for the investment made by my own parents, Bob and Juanita Simpson. They provided all the essential elements of an enriched childhood environment; especially love and attention - the prime ingredients. Thanks Mom, for your willingness to type papers during those early quarters. That investment not only helped me meet my course requirements, but also gave me more time to spend with Laura and the kids. Thanks to you both for letting me room at home, when I shipped Laura and the kids ahead to Virginia. Those months of monastic isolation were absolutely critical to completing my work, but without your collective caring for my creature comforts, I would never have been able to survive (emotionally or physically). For this and so very much more, I owe you both my eternal devotion.

To my thesis advisor, Janet Kolodner, I owe a special thanks. Janet introduced me to AI and was extremely patient with me as I struggled to understand AI research. She clearly has made a major investment in this dissertation. Without her inspiration and advice, I do not believe I would have found the determination to carry my research through to completion. Janet, as your first student, I hope I can be a credit to you. Now that I've broken the ice, I hope your other students are not nearly as difficult as I was.

My thesis committee, Andy Smith, Andy Spiessbach, Larry Barsalou, and Bill Underwood, deserve thanks for their patience and support. I am indeed fortunate to have had a committee composed of such learned and gracious gentlemen. Thank you all for your investment. A special thanks goes to Andy Spiessbach for being a considerate friend and letting me blow off steam, when things seemed out of hand. Thanks Andy, I needed that.

There was one other member of my original reading committee, Jim Gough, who passed away before I finished my research. Jim was a genuinely nice man. All of us who knew him were enriched by his good humor and grace. He was the kind of person who never met a stranger. Even when his health was failing badly, he would offer to read my drafts and talk about my research. I'm proud to have known Jim and thankful for his investment in me.

Friendships are one of the happy byproducts of being a PhD student. Among those students who have helped make this process more tolerable are: Katia Sycara-Cyranski, Stefan Wrobel, Dana Eckart, Kirt Pulaski, Gene Spafford, Tom Wilkes, and Arnold Robbins. Katia has been an especially helpful as a sounding board for ideas, proof reader, copy editor, and dear friend. Stefan and before him Dana have provided the necessary support of the Symbolics software that made research possible. Dana made programming contributions that were extremely useful in dealing with "Flavors." Kirt was a very good proof reader and helped wring out many typos and misspellings. Gene and Tom were always helpful in dealing with other ICS computers and could be depended on for sound advice. Arnold was a life saver on several occasions in keeping the old software tools formater alive until I finished writing this dissertation.

A couple of other institutions deserve my thanks. First, Georgia Tech will always remain special to me. I suspect few graduate schools would have admitted me with as little academic evidence as I had to offer. Tech did admit me, based on faith alone I suspect. I hope to do Ma Tech proud as an alumnus. Second, I also want to thank Symbolics for building the LISP Machine. Without the environment and support tools made available by that company, I doubt I would have been capable of nearly as much research.

Finally, I would like to recognize the important conceptual support provided by the Harvard Negotiations Project, which put out such books as Getting to Yes by Fisher and Ury and The Art and Science of Negotiation by Raiffa. Without such well thought out works on negotiation I would never have been able to capture the domain knowledge needed to demonstrate case-based reasoning in the dispute mediation domain. Also my thanks to Sherman Dallas of the Tech College of Management for pointing out the difference between negotiating, arbitrating, and mediating.

In the text of this dissertation, I use the plural pronoun "we" frequently. I do this in recognition of the fact that this research would not have been possible without the love and support of all those mentioned above. The errors and oversights are surely my own, but if this research is viewed favorably, then it is in large measure due to my collective investors. Thanks to you all!

TABLE OF CONTENTS

CHAPTER I

CASE-BASED REASONING IN PROBLEM SOLVING

> "In order to solve a new problem one uses what might be called the basic learning heuristic - first try using methods similar to those which have worked, in the past, on similar problems." (Minsky, 1963)

1.1 Introduction

In the course of everyday problem solving, people often recall past experiences with similar problems to guide their reasoning actions with respect to their current problem. The following examples illustrate this process:

> A lawyer listening to a client describe his case is reminded of a legal precedent that he had previously used as the basis for another client's defense. He considers whether it is also applicable to the current case.
>
> A doctor notes that this patient's symptoms are reminiscent of an unusual case that he had once misdiagnosed. He diagnoses it correctly this time.
>
> An investor recalls that the last time the the prime rate fell, the stock market rose sharply and he had lost an investment opportunity. He rushes to make his investments immediately.
>
> An algebra student, contemplating a homework problem, remembers that the teacher had worked out a similar problem in class. Guided by the worked out example, the student sees how to solve the homework problem.
>
> A babysitter decides that a good way to resolve a squabble between her two charges is to use a technique she remembers her mother had employed when she had a similar fight with her sister. Peace is soon restored to the play room.

These examples illustrate a type of problem solving which we call *case-based reasoning*. In case-based reasoning, a current problem is resolved by analogy to a similar past experience or case. The knowledge used or decisions made in a previous case serve as heuristic advice in reasoning about how to solve a new problem. This thesis investigates the use of case-based reasoning in the design of computer problem solving systems.

Current approaches to problem solving in artificial intelligence (AI) have failed to achieve human levels of performance except in well understood, highly constrained situations. In general, only those situations which can be handled by static algorithms and prepackaged knowledge can be solved by current reasoning methods. This limitation in current approaches is due primarily to three common design decisions:

1. Current problem solving systems are designed to solve each problem from scratch. Their computational lines of reasoning are static and often extremely long; even for repetitions of the same or similar problems.

2. Current problem solving systems are not designed to learn. Direct human intervention is required to optimize programs for recurring types of problems and repair errors in reasoning that lead to failures.

3. Current problem solving systems are usually designed with separate functional modules that make the integration of multiple lines of reasoning extremely difficult. This tradition has viewed learning and analogical problem solving as separate, isolated types of reasoning. In the absence of an integrated view, the constraints imposed by these components on each other, as well as the assistance available to each from the others, has been largely ignored.

Each of these flaws points to a need for more flexible, adaptive reasoning systems that can automatically adjust to the problem solving environment. Providing a problem solver with capabilities of accessing a memory for experience and reasoning analogically from previous cases allows the problem solver to focus on only the relevant parts of a new problem, avoid past failures, and possibly resolve the problem more efficiently. The process which we call case-based reasoning is one methodology for providing problem solvers with such adaptability. Case-based reasoning exploits repetition in problem solving by storing the results of its

computational decisions in a dynamic memory that is integrated with all problem solving processes. The results of previous decisions are analyzed and stored along with each case in the memory. These cases then become available to guide and direct later problem solving.

If automated problem solvers were to have access to previous experience as a source of heuristic advice in the analysis and solution of new problems, they, like people, would be capable of automatically changing their behavior by analysis of previous experience. This requires a problem solver design which integrates problem solving, learning, and analogy. With this objective in mind, the approach to case-based reasoning includes the following two major design decisions:

1. *A conceptual memory for experience is integrated with problem solving processes and accessed by the problem solver.* Analogy to similar cases during problem solving offers the potential to reduce both the number of problem features that must be investigated and the number of reasoning steps necessary to reach a solution.

2. *The facilities for feedback and evaluation are integrated with problem solving.* This allows the problem solver to learn from its experience. Success biases the problem solver toward repetition of previous decisions. Failures bias the problem solver away from faulty decisions.

People seem to do analogical reasoning as a natural part of their problem solving. Much psychological evidence indicates the importance of analogical problem solving in diverse areas of human experience (Clements, 1981, 1982; Gick and Holyoak, 1980, 1983; Luchins, 1942; Reed et al., 1974). It seems especially useful in ill-understood domains or in the early phases of skill or knowledge acquisition (Anderson et al., 1984; Chi et al., 1981; Ross, 1982).

Our research provides two significant advances that can lead to improved computer problem solvers. First, we present an integrated process model and demonstrate how case-based reasoning can support problem solving. This model is implemented in a computer program called the MEDIATOR that offers advice for the common sense resolution of disputes. It does this by employing case-based reasoning to resolve disputes. As a result of analysis of its behavior, it incrementally changes its reasoning. Second and more generally, we provide a way of designing more flexible problem solving systems which can store and recall their experiences, assess their performance, and modify their later behavior accordingly.

1.2 Capabilities and requirements for problem solving

Case-based problem solving is a process of using decisions made in similar situations to suggest a means of dealing with a new problem. The hypothetical case below shows the use of case-based reasoning during several different problem solving tasks.

SINAI DISPUTE

A mother reads in the paper about the Sinai dispute (before the Camp David Accords). She is reminded of the Korean War since both are disputes over land, both are competitive situations in which the conflict can not be resolved completely for both sides, and in both, military force had been used previous to negotiations. Based on this reminding, she predicts that Israel and Egypt will divide the Sinai equally, since that is what happened in the Korean War.

She later reads that the USA had suggested this solution and it had been rejected by both sides. She is reminded of her daughters' quarrel over an orange. She had suggested that they divide it equally, and they had rejected that, since one wanted to use the entire peel for a cake. Realizing that she hadn't taken their real goals into account, she then suggested that they "divide it into different parts" -- one taking the peel, the other the fruit. This reminding provides the suggestion that failures may occur because the goals of the disputants are misunderstood. She therefore attempts a reinterpretation of Israel's and Egypt's goals. By reading more closely, she learns that Israel wants the Sinai as a military buffer zone in support of national security, and Egypt wants the land back for national integrity.

She is reminded of the Panama Canal dispute since the disputants, disputed object, and goals are similar to those in the newly understood Sinai dispute. In that case, the USA returned economic and political control of the Canal to Panama, but retained military control for national security reasons. Analogy to that incident leads the mother to decide that a similar division of the Sinai would be reasonable and guides the refinement of the "divide into different parts" plan. Replacing the US by Israel (the party currently in control of the object) and Panama by Egypt (the party who used to own it and wants it back), she predicts that Egypt will get economic and political control of the Sinai, while its normal right of military control will be denied.

By making reference to previous similar cases, this problem solver has been able to understand a new problem and make predictions about its outcome by using plausible inferences transferred from those cases. Because the mother chose to view the Sinai dispute as an analogy to the Korean War, for example, she could quickly estimate a potential outcome for the

dispute and avoid a lengthy static investigation and evaluation of possible alternatives. When the mother later realized her predictions were in error, other cases were used to explain the earlier failure and provide alternative predictions. For example, by basing her reasoning on the Panama Canal dispute, the mother was able to offer an alternative prediction. This tells us the following about the inclusion of case-based reasoning in problem solving systems:

Problem Solving Principle #1

Including a capability for case-based reasoning in a problem solving system allows previous computations to be used to suggest solutions to new problems, potentially cutting down the work required to solve a difficult problem from scratch.

The inclusion of case-based reasoning in our problem solving systems forces several requirements on those systems. First, we must consider where the previous cases come from. It seems reasonable that the mother, in the example above, had stored her experience in memory in such a way that her current situation could be used as a cue in its recall. The storage and recall of experience is such a natural part of our own cognitive processes that people often fail to take notice of them. If we expect computer problem solvers to refer to previous similar cases during problem solving, then we must provide them with the capability to store those cases in an experiential memory and retrieve them at the appropriate time. This provides our next principle:

Problem Solving Principle #2

Case-based reasoning requires access to a dynamic memory capable of storing and retrieving previous experience.

Next, we must consider which previous cases our problem solver should remember. Some mechanism is necessary to insure that only cases potentially relevant to resolving a current case are made available to the problem solver and of these cases only a small number are actively considered. Otherwise, the problem solver would be overwhelmed by the number of potential analogies.

At different times during her reasoning about the Sinai dispute, the mother, as our hypothetical reasoner, actively considered three different cases: the Korean War, the orange dispute, and the Panama Canal dispute. The fact that the focus was on only three cases out of possibly thousands in the reasoner's memory indicates a capability of noticing relevant similarities of concepts. The Korean War case, while different from the Sinai dispute shares several important features with it that promote its retrieval as a similar case: both are disputes over land and both involved the use of military force. Because only similar cases will help in doing case-based reasoning, we put the following requirement on case-based problem solving.

Problem Solving Principle #3

Case-based reasoning requires that a problem solver be able to recognize similarity between cases so that only those potentially applicable to the current problem are recalled.

We shall see in later sections that a memory organization based on abstraction of similarities and indexing by differences allows this to happen.

It is reasonable to suppose that at the same time the mother originally recalled the Korean War case, she was also reminded of the Panama Canal dispute. This is reasonable, since it too shares many of the same features with the Sinai dispute. Given that memory may provide many cases similar to a case being considered, a selection process is necessary to choose from among those cases the one or the few which can potentially provide the best advice. This judgement requires a relative ordering of items already judged to be similar to a current situation. Based on this observation, we state the following principle:

Problem Solving Principle #4

Case-based reasoning requires choosing the most appropriate case from a set of potentially applicable ones.

We employ an ordering process that assigns a weight to each feature type in a dispute. Alternative cases are then evaluated according to a series of elimination and ranking tests.

Once a previous case is chosen, some portion of it is transferred for use in resolving the new situation. In the example above, the reasoner used the outcome of the Korean War as a means of predicting the likely results of the Sinai dispute. After that prediction failed, she transferred an explanation for the failure from a previous similar failure. After correcting her misunderstanding, she predicted a new outcome for the Sinai dispute by transferring knowledge from still another recalled case. Based on this observation, we are led to the following requirement:

Problem Solving Principle #5

Case-based reasoning requires that the problem solver be able to transfer the appropriate information from one case to another.

As we shall demonstrate, the specific decisions that a problem solver needs to make at different points in the process guides the selection of the appropriate information to be checked and possibly transferred from previous cases.

Once again referring to our earlier example, we notice that the mother was reasoning analogically from cases that were not only similar in terms of the encountered problem, but ones which had led to what she believed to be successful problem resolutions. How did she determine those cases were successes? If, in the case of the daughters' quarrel over the orange, the sisters had stopped their quarrel, it seems reasonable that the mother interpreted the end of the quarrel as a kind of success signal. More generally, this implies that the mother had received feedback allowing her to assess her previous problem solving performance. Successful resolution of one problem using a particular plan encourages a problem solver to employ the same reasoning in future cases. If the mother had not been able to find some means of correcting her reasoning (explain and remedy her failure) in her daughters' case, there would be less inclination to adopt the same reasoning for a similar failed case such as the Sinai dispute. Evaluation of success or failure is an important requirement for problem solvers that are designed to adapt to their environment. We cannot expect a problem solver that never knows the results of its suggestions to modify its behavior. This leads us to conclude that any model of problem solving must include the following characteristic:

Problem Solving Principle #6

Case-based reasoning requires that the problem solver must receive feedback and be able to evaluate its decisions.

It is as a consequence of this principle that the model of problem solving that we present in the next section explicitly includes feedback and evaluation components.

It follows from principle #6 that if we enable problem solvers to evaluate their decisions, then we must provide them with the capacity to recover when they decide that they have failed. One of the notable aspects of the reasoning used in the example above is that the reasoner initially failed, but was able to determine a reasonable explanation for her initial failure and recover successfully. Problem solving in the absence of perfect knowledge is likely to lead to failures. This leads us to conclude the following:

Problem Solving Principle #7

Problem solvers must be able to recover from reasoning failures.

Our approach to error recovery is to view it as another instance of case-based reasoning. This approach is reflected in our process model that is presented in the following section.

In summary, including case-based reasoning in problem solving forces many requirements on the problem solving system. It must be able to remember past cases, judge which of those are the most applicable for use in evaluating a new case, transfer knowledge from one case to another, and evaluate feedback on its decisions. In the following sections, we outline a problem solving model and a memory organization that allow these things to happen.

1.3 A model of case-based problem solving

The process model that supports case-based reasoning integrates problem solving, learning, and analogy. The problem solving framework includes four problem solving tasks: problem understanding, generation of a plan to resolve the problem, evaluation of feedback to determine success or failure, and failure recovery in the latter situation. Learning is integrated within this framework via a dynamic memory which remembers problem solving cases and makes them available for later problem solving. Analogy is considered in its roles in the problem solving framework: previous similar experiences are located and retrieved from memory, the most appropriate ones are selected from those retrieved, and information is transferred from the previous cases as required by the various problem solving tasks. Figure i-1 below shows the basic problem solving framework. Case-based reasoning, including memory access, is a part of each process. We overview each part of the framework below.

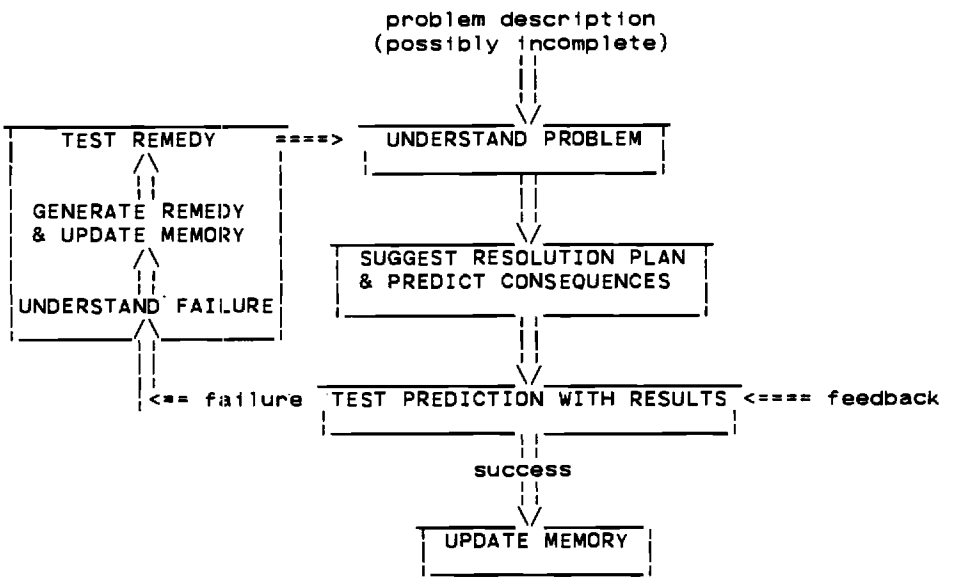


Figure 1-1

The first task, *problem understanding*, receives the initial problem description and constructs an internal representation of the problem. Two important stages of this task are problem classification, where more specialized categories are identified for the problem, and elaboration, where missing information is inferred to complete the representation. Case-based reasoning provides heuristic support for the problem understanding task by examining portions of similar cases and providing:

1. plausible categories as part of problem classification.

2. plausible information to fill in missing parts of the new representation.

For example, the reasoner in the Sinai dispute remembered the Korean War and decided that, like the recalled case, this was another "possession dispute between polities." Using this classification, a reasoner can make other decisions consistent with this category.

After a representation of the problem has been constructed, the *planning* task is responsible for generating a solution to the problem. This includes making decisions about how the planning should be done, selecting and refining appropriate plans, and predicting the consequences of the plan's employment for a particular problem. During planning, cases made available from memory enable the case-based reasoning process to provide:

3. suggestions for how the planning process should proceed.
4. recommendations supporting the use of a particular plan.
5. recommendations against the employment of a certain plan.
6. suggestions for plan refinement for this specific case.
7. predicted outcome of the selected plan in this situation.

In the Sinai dispute, for example, the reasoner used the results of the Korean War to predict both the type of plan that should be employed (i.e., "Divide Equally") and the likely outcome (i.e., both sides will get half of the Sinai).

In the next stage of our problem solving framework, the predictions (provided by the planning stage) are tested against the results received as feedback from plan application. If the predictions hold, then the case is stored in memory as a new successful problem solving experience. If the predictions are violated, then a failure is recognized and recovery is attempted.

During *failure recovery*, an explanation for the failure is determined and an appropriate remedy selected. If the remedy eventually allows successful resolution of the original problem, then the entire sequence of attempt, failure, remedy, and final success is stored into memory. During failure recovery previous cases and case-based reasoning can provide:

8. explanations for the failure in the current case.
9. suggested remedies to correct the failure.

Because our framework, unlike most other problem solving models, explicitly deals with failure, we will briefly describe this process in more detail. Failure requires a problem solver to rethink problems and come up with new solutions. This requires figuring out what went wrong, perhaps reinterpreting the problem, and coming up with new resolution plans. This entire process is referred to as either failure recovery or *remediation*. Recovery from failure is viewed as another instance of problem solving within our problem solving framework. Whereas a problem solver originally had to understand the problem, he now must *understand the failure*. Previously he had to suggest resolution plans, now he must select a remedy for an identified error in his reasoning that is believed to have caused the failure. When remediation is done, the problem solver is ready to try once again to resolve the original problem. Because remediation deals with and reasons about the problem solving process itself, it is sometimes referred to as *meta-problem solving* (Hayes-Roth and Hayes-Roth, 1979; Stefik, 1981; Wilensky, 1983).

The first stage in failure recovery, as another instance of the problem solving process, is understanding the error. This requires that a problem solver know how the solution was developed, what inferences were made, and what kinds of errors were possible. This type of knowledge, sometimes called meta-knowledge, deals not with the actual problem domain, but with knowledge about problem solving. Looking at the problem solving model presented above, we know that errors are possible anywhere a heuristic decision has been made. Such decisions are made during understanding and in plan selection, for example. In the Sinai dispute example presented above, the reasoner assigned blame for her failed prediction to a misunderstanding of the disputants' goals in the original problem.

After a problem solver has understood the failure, the second stage in failure recovery is the selection and application of a remedy. Remedies are associated with each type of resolution failure. For example, once the reasoner has decided that her failure in the Sinai dispute was due to a goal misunderstanding, she repairs this error by the application of a remedy that seeks to identify alternate goals and makes the appropriate change to her internal representation of the case. With this change, the reasoner can reprocess the problem and produce an acceptable solution.

Note that problem solvers in this situation must actually make three separate problem solving passes. The first pass is a resolution attempt that fails, the second pass is a failure recovery attempt (remediation) that alters the problem representation, and the third problem solving pass reaccomplishes the problem solving which finally succeeds.

To indicate how case-based reasoning fits into the model shown in Figure 1-1, we add two memory processes: update and retrieval. Memory update implements basic learning mechanisms so that cases may be stored for later use. Two different types of cases are stored, one group reflects those cases which required no error recovery, the others involved the additional reasoning accompanying failures and their analysis. Retrieval then operates on the stored set of cases to make the appropriate ones available to the three problem solving tasks: understanding, planning, and failure recovery. The complete process model of case-based problem solving is indicated by Figure 1-2 below:

CASE-BASED PROBLEM SOLVING

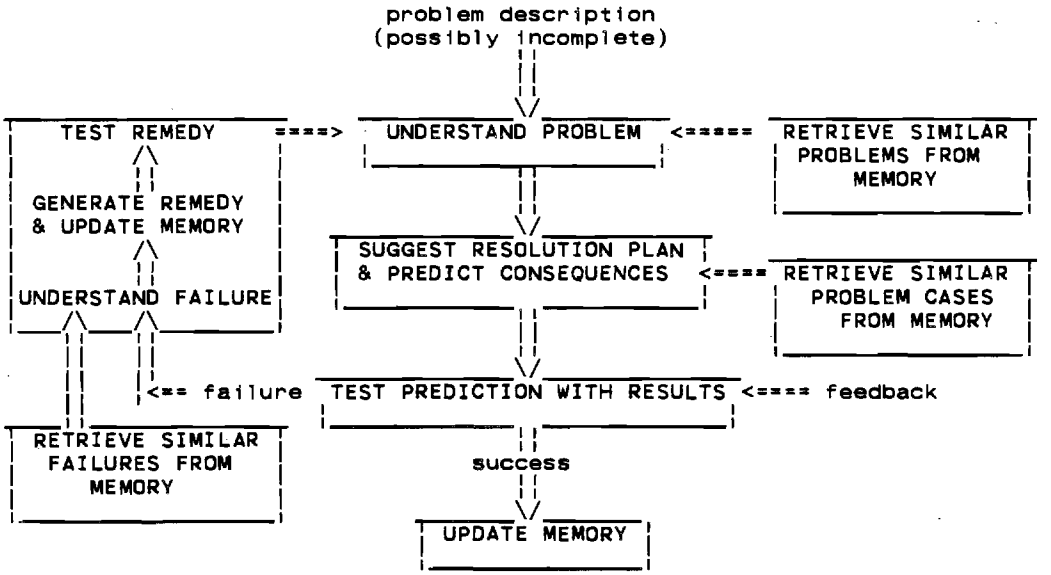


Figure 1-2

Chapters three through five address specific technical details of the processes outlined above.

1.4 A dynamic memory for cases

In chapter six, we will discuss the details of dynamic memory that are necessary to support case-based reasoning. Rather than delaying all discussion of memory until then, however, we present in this section an overview of the basic ideas of *dynamic memory* (Kolodner, 1984; Schank, 1982) that will provide the fundamental ideas upon which later chapters will depend.

Case-based reasoning represents an attractive approach to problem solving because of its potential to replace a lengthy computation by the retrieval and transfer of a previous similar computation. For this potential to be realized, the cost of storing, retrieving, and transferring the information must be less than the cost of its recomputation. It is for this reason that the problem solver's memory is such a crucial component. Retrieval can be made very rapid if the information is organized effectively (Aho et al., 1974). Thus the organization of cases in memory must consider the effects on their retrieval. In this section, we will discuss an approach to organizing cases in a dynamic memory such that the following requirements are satisfied:

1. *Cases can be retrieved based on conceptual similarity.* This enhances the chances of retrieving a potentially applicable case when faced with new or unexpected problems.

2. *Retrieval of cases does not slow appreciably as new cases are added to memory.* This is necessary to insure that case-based reasoning remains a cost effective alternative to recomputation.

3. *Retrieval is directed by the concept being sought, not by any special knowledge of the memory organization.* This restriction is intended to prevent a retrieval process based on blind search of memory categories.

4. *Retrieval will always return only the most similar cases in memory.* We want memory to always return at least one potentially applicable case if there is one, while screening out as many cases as possible.

1.4.1 Conceptual representation and memory

A computer memory whose information is organized by conceptual similarity is known as a *conceptual memory* (Kolodner, 1984). The basic idea of conceptual similarity, which we will make more precise in chapter six, can be illustrated by comparing the common sense concepts "orange" and "candy." While lexically dissimilar, both are conceptually quite similar. Both are specialized concepts of the more general concept "food." It is reasoning based on this type of conceptual similarity that needs to be employed during case-based problem solving.

In a conceptual memory, information is organized and retrieved by concepts. So information about "disputes," for example, is organized around the concept "dispute." This allows different lexical symbols such as "quarrel," "fight," or "squabble," which reference the same concept, to be organized together in a single conceptual "dispute" memory organization. Since we are interested in the meanings and not lexical symbols, we must represent concepts of interest in terms of a consistent set of primative representations. The approach to conceptual representation employed in this research was motivated by the theory of conceptual dependency (Schank, 1972).

To conceptually represent a problem such as the "orange dispute" where a mother encounters her daughters' engaged in a quarrel over an orange, we must first identify the salient conceptual components of the problem (i.e., "disputes") in the abstract. For example, the daughters, their verbal exchanges, and the orange all fill specific required roles within an abstract "dispute" concept. We can identify these roles as being the "disputants" roles, the "arguments" roles, and the "disputed object" role. Using this approach, we can impose a conceptual structure on problems such as the orange dispute. Figure 1-3 presents a conceptual view of the orange dispute as an instance of the abstract concept "dispute:"

A CONCEPTUAL REPRESENTATION OF THE "ORANGE DISPUTE"

```
PROBLEM:  "DISPUTE" with
              name: orange-dispute
              disputant1: sister1
              argument1: wants possession of orange1
              disputed-object: orange1
              disputant2: sister2
              argument2: wants possession of orange1
```

Figure 1-3

A representation such as Figure 1-3 imposes a standard structure on a concept. This allows us to reason uniformly about similar concepts, for example, to determine its attributes or characteristics. Once we identify that the above concept is a "dispute," we can anticipate that certain characteristics can be accessed and determined. For example, by accessing the arguments within this representation, we can characterize the orange dispute as one where the disputants both wanted possession of orange1 (i.e., the disputed object). We need to be able to characterize concepts this way in order to (1) relate similar concepts to each other in larger groupings of concepts and (2) differentiate similar concepts from each other. We refer to a larger group of concepts as a *conceptual organization* (Kolodner, 1984) if it permits similar concepts to be collected together such that they can be differentially retrieved when necessary. Intuitively, this implies that we want to organize, for example, all dispute cases around the "dispute" concept, while insuring that new cases can be added in the future and old cases can be retrieved by their distinctive characteristics. Thus, even though dispute cases like two men squabbling over a window and two little boys fighting over a candy bar should be grouped together because they share the same underlying concept, they should still be distinguishable by their differences (e.g., the disputed object is a window in one and a candy bar in the other).

Just as reasoning about individual concepts is simplified by the use of conceptual structures, the reasoning associated with larger groups of concepts within a conceptual memory is simplified by the use of a *memory structure* which contains information about the concepts grouped within it. The memory organizing structure used in this research is based on the idea of *generalized episodes* (GE)* (Kolodner, 1984; Schank, 1982). Generalized episodes provide a unified approach to the problems of organizing a conceptual memory according to our requirements. Specifically, generalized episodes allow the following:

1. organization of domain concepts,
2. retrieval based on conceptual similarity, and
3. integration of new cases into the existing memory.

1.4.2 Organizing concepts in memory

Generalized episodes organize cases into a network where each node is either another generalized episode or a specific case. Generalized episodes have two components: (1) the *norms* of the generalized episode which represent the abstracted content of all the cases organized within that particular episode and (2) the *indices* which connect the generalized episode with the tree of other generalized episodes and specific cases organized below it. Figure 1-4 below shows the abstract structure of a simple generalized episode. The norms of the generalized episode are contained in the upper portion of the diagram. The indices are shown below the norms and are labelled to illustrate how the different cases, which are located at the leaves of the tree, can be distinguished.

ABSTRACT STRUCTURE OF A SIMPLE GENERALIZED EPISODE (GE1)

norms: The norms part of a generalized episode contains abstract general information that characterize the cases organized below it. It represents, in a compact form, a general "prototype" or abstract view of some specific aspect of the individual cases.

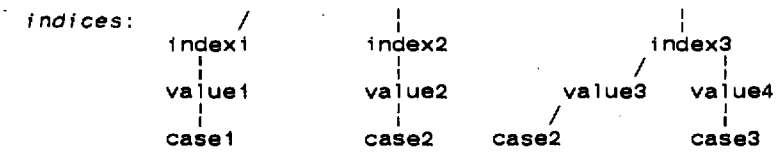


Figure 1-4

Specific cases are accessible from the top of a generalized episode by travelling across the labeled arcs which connect the case to the norms. These labeled indices serve to differentiate specific cases from those cases that are "normal" and thus typified by the features described in the norms. Notice that the arcs connecting cases to the norm have two labels. The first label is an index type (e.g., index1). The second label is an index value (e.g., value1). We can access case1, for example, by travelling across the arc labeled by index1 and value1. Notice that case1, in this example, is accessible by only this single path. Case2, however, is accessible via two different paths involving different indices and values (i.e., index2 with value2 or index3 with value3). Also notice that case2 and case3 share one index type (index3), but are differentiated because they have different index values (value3 and value4).

---

*Generalized episodes are related to Schank's (1980) MOPs, Kolodner's (1984) E-MOPs, and Lebowitz's (1980) S-MOPs. The term generalized episode is used to avoid any confusion that might be caused by my variation from the technical details of these specific memory structures.

To illustrate how generalized episodes organize dispute cases, consider the following two dispute problems:

TWO DISPUTE PROBLEMS

PROBLEM1: "DISPUTE" with
name: orange-dispute
disputant1: sister1
argument1: wants possession of orange1
disputed-object: orange1
disputant2: sister2
argument2: wants possession of orange1

PROBLEM2: "DISPUTE" with
name: window-dispute
disputant1: man1
argument1: wants window1 open
disputed-object: window1
disputant2: man2
argument2: wants window1 closed

Figure 1-5

Both of these problems are disputes between people interested in some physical object. One dispute is over possession of an orange (PROBLEM1), and the other is over the position of a window as indicated by the arguments. One of the generalized episodes used by the MEDIATOR is "physical disputes." Certain dispute problems faced by the MEDIATOR are organized within this generalized episode; specifically those in which the disputants' goals involve either the use or possession of a disputed object. Most of the cases in the MEDIATOR's experience involve disputes between people over possession of some physical object, so this is considered the norm for physical disputes. The two problems above are differentiated in the MEDIATOR's "physical dispute" generalized episode as shown in Figure 1-6:

THE "PHYSICAL DISPUTES" GENERALIZED EPISODE

norms:
dispute is over possession of object
object is a kind of (ako) physical object
disputants are people
disputant's goals are physical control goals
precedent case is orange dispute

indices:

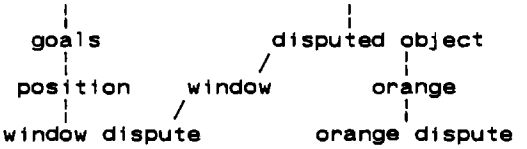


Figure 1-6

The norms, as illustrated in Figure 1-6, reflect the general abstract content of the cases it organizes. The norms in this research also include one other component that does not reflect cases in the abstract. This component is the *precedent case*. A precedent case is the special exemplar that is associated with a generalized episode. Usually it is the specific case that caused the generalized episode to be created. In the example above, the precedent case is the orange dispute. The precedent case can always be accessed from a generalized episode since it is contained within the norms and does not require travelling across any indices. The inclusion of a precedent case within the norms of a generalized episode is one technical difference between this and previous research.

In the example above, the orange dispute is differentiated from all other physical disputes by the fact that its disputed object was an orange. The concept orange is said to *index* that dispute within the physical dispute generalized episode. Also notice that the orange dispute is not indexed by the goals of the disputants. This is because their goals were the same as the norms for physical disputes (i.e., possession goals) so this case need not be indexed by the disputants' goals. The orange dispute case is thus relatively typical of other physical disputes because it had very few distinguishing features that would cause its specific retrieval. Compare it to the window dispute, also indexed within the same generalized episode. The window dispute is more atypical, and thus is indexed in more ways within the generalized episode.

As a new case is entered into a generalized episode, the features that differentiate it from the norm are extracted and used to create new indices that will then point to the new case. If another case is already indexed by the feature, a new generalized episode is formed. The similarities are used to build its norms and the differences are used to index the cases within the new episode. To retrieve a case from a generalized episode, the features of the new case are specified, the indices corresponding to those features are traversed, and the case indexed by those features are then available for retrieval.

As indicated above, individual cases are indexed by component features that distinguish them from other cases in the generalized episode. One feature that warrants special emphasis is failure. When the problem solver fails to resolve a problem, the case is also indexed in memory by the failure. This allows learning and reminding on the basis of failure. If blame can be assigned for a failure, the case is indexed by those features which caused the failure. For example, the problem solver may determine that he attempted to resolve the problem using a bad plan. So the features of the problem are used to index the failure as well as the failed use of the particular plan. When a second similar situation is encountered, these features serve as an index to a failed case. If a solution was found to the first failure situation, it can be applied to the second so that if failure cannot be avoided, error recovery can be better directed.

When a generalized episode has only one case with a certain feature, the index for that feature will be sufficient to retrieve the individual case. This is the situation for the simple "physical disputes" generalized episode above. If two or more cases share a common feature, the index for that feature will point to another generalized episode, with the same structure, that organizes this specialized subset of cases. Its norms will come from the similarities between the subset of cases it organizes. Using the same abstract view of generalized episodes as before, suppose another case were added which resulted in a new generalized episode being formed. The resultant change in the generalized episode, GE1, is illustrated in Figure 1-7 as a new specialized episode labelled GE2:

ABSTRACT STRUCTURE OF A COMPLEX GENERALIZED EPISODE (GE1)

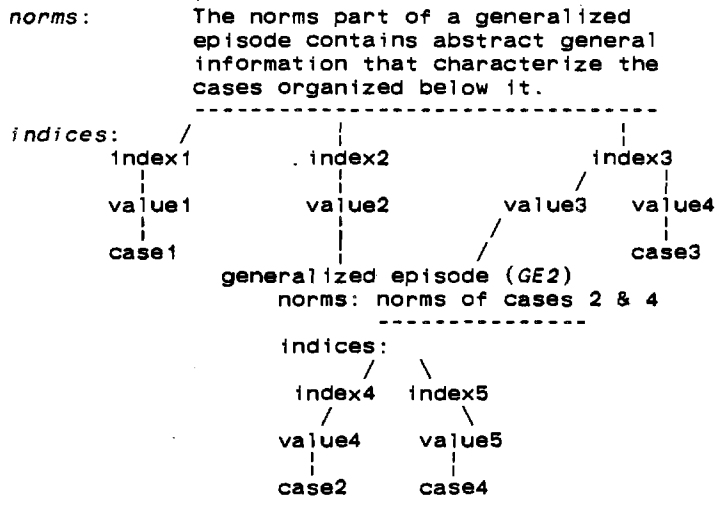


Figure 1-7

This change can be illustrated in the "physical disputes" generalized episode introduced above. Consider the following additional dispute problem that will need to be added to the previous two:

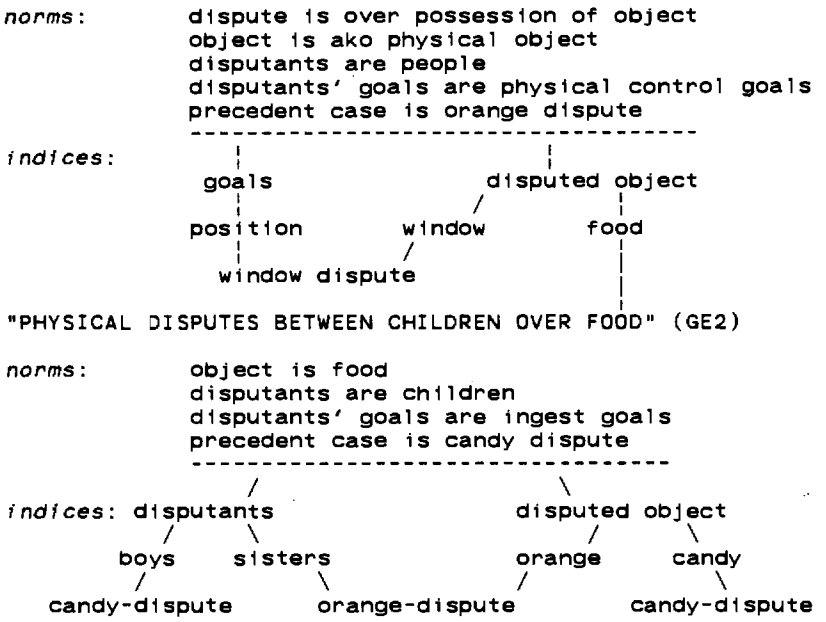
ANOTHER DISPUTE PROBLEM

PROBLEM3: "DISPUTE" with
name: candy-dispute
disputant1: boy1
argument1: wants possession of candy1
disputed-object: candy1
disputant2: boy2
argument2: wants possession of candy1

Figure 1-8

This new problem is another dispute, this time involving two boys fighting over possession of a candy bar. Because this dispute is very similar to the norms for "physical disputes," little change will result from its inclusion in the "physical disputes" generalized episode. One noticeable change, shown in Figure 1-9, is in the indices organizing disputes according to the features of the disputed object. Because the orange and the candy are both food, this becomes part of the new norms for the new generalized episode (GE2) organizing these two cases. This illustrates how lower level generalized episodes organize increasingly more specialized concepts.

THE CHANGED "PHYSICAL DISPUTES" GENERALIZED EPISODE (GE1)



"PHYSICAL DISPUTES BETWEEN CHILDREN OVER FOOD" (GE2)

Figure 1-9

As this example demonstrates, the common characteristics of the orange dispute and the candy dispute (e.g., object is food) has been captured in a new generalized episode (GE2) indexed off the original episode (GE1). GE2 is a specialization of the generalized physical dispute episode organizing cases dealing with children quarreling over food. Individual cases are still accessible via the distinguishing features of the disputants and the object within GE2.

1.4.3 Retrieval in conceptual memory

An organization such as that provided by generalized episodes provides numerous cross-indices for cases that differ from the norm in several ways. This allows retrieval of a similar case to occur via several different paths. Given this organization, retrieval of cases need not be a blind search, but can be directed to specific generalized episodes. This is the subject of this section.

Case-based reasoning depends on the retrieval of potentially applicable cases from memory at those points in the problem solving process where a problem solver needs to make heuristic decisions. The organization described above provides a way of locating exemplars to use in reasoning about a new case. The retrieval process which allows similarity-based "reminding" is a traversal procedure. When a new case is encountered, its features act as cues for each generalized episode associated with components of the problem. Links associated with each cue are traversed so that the generalized or individual episodes most similar to the case are found. It is these cases which are now available for further evaluation. For example, to retrieve a "physical dispute over an orange" in the "physical disputes" generalized episode shown in Figure 1-9, the "disputed-object" index labeled with the value "food" would be followed to find, in this instance, another generalized episode (GE2). Next the "disputed-object" index corresponding to "orange" would be followed to find the orange-dispute case indexed at that point.

The organization of cases using generalized episodes, as illustrated above, leads to a richly cross-indexed memory. A retrieval process based on blind search would run the risk of either searching the wrong generalized episodes, requiring an excessive retrieval time, or being cut off before finding the most applicable cases. For this reason, retrieval is constrained to be a *directed search* rather than an unconstrained search typified by the usual notion of "spreading activation" (Quillian, 1968). Using a directed search means that the

retrieval process can only traverse an index corresponding to specified features provided by the cue. One problem in retrieval is the specification of the appropriate index for traversal. This is known as *index selection* (Kolodner, 1984). Index selection is important not only during retrieval, but also during the process of adding a case to memory. A consistent means of index selection minimizes the chance of storing an irretrievable case in memory.

In the earlier retrieval example of finding a "physical dispute over an orange", a specific case was located as a result of the memory traversal. This was made possible by the fact that the retrieval cue specified a feature that was both indexed in the memory organization and unique to a single case. This is clearly not always possible, especially for problem solvers who rarely encounter "exactly" the same problem twice. For example, what if a problem solver next encountered a "physical dispute over a cookie". The same traversal process would arrive at the generalized episode that organizes both the candy and orange disputes, because the cookie is also "food." But at this point, there are no indices that correspond to the concept "cookie". The traversal process has found a general, not a specific concept. This happens when an unindexed feature is encountered, as happened here, or when the retrieval cue is too general. For example, we would have the same difficulty with a "physical dispute over a piece of food." Because retrieval is a directed process, some method is needed to allow retrieval to continue in these situations. In the design of a retrieval process, there are three options at this point:

1. elaborate the retrieval cue,
2. return a generalized concept, or
3. return a precedent case.

The option to elaborate the retrieval cue was demonstrated by Kolodner (1984). It involves the use of knowledge about the cue to infer plausible values for unspecified parts of the representation. Alternatively, the generalized concept located at that place in memory could be returned. This option may prove useful in some specific situations. For example, it could support some problem solving decisions where the consequences of previous such decisions, in general, may help the problem solver choose between alternatives. When the retrieval process is attempting to retrieve a case however, the return of a generalized concept will not prove helpful. Therefore another solution has been found useful in this situation. This involves using the precedent case. The precedent case is always retrievable from a generalized episode, when further traversal is impossible. For the example above, the candy dispute case would be returned as an exemplar of the probe into memory for a dispute over food. As a consequence, when the retrieval process can no longer specify indices for traversal, the precedent case allows a specific reminding to be returned from a probe into a generalized episode.

The combination of extensive indexing of cases along with the default retrieval of precedent cases means there are likely to be many remindings caused by processing a new case. This is a desired feature, since we want to ensure recall of any useful previous experience that might reasonably aid problem solving. We will discuss the problem of choosing the most applicable case from all those that have been recalled in chapter six.

1.4.4 Adding new cases to memory

As new cases are encountered and added to memory, it is important to maintain the proper organization. The memory update process is responsible for insuring that a new case is indexed into memory in such a way as to be accessible by the retrieval process just described. Memory update proceeds systematically to each generalized episode associated with the components of the new case. For each generalized episode, features of the new case are used to index it properly into memory.

There are three possible consequences of indexing the new case by a certain feature (Kolodner, 1984). First, if the feature is new, a new index is constructed linking the new case to the generalized episode via this feature. Second, if there is another case indexed by that feature, then a new generalized episode is created with the similarities between the two cases becoming the norm and the specific cases are indexed off this new episode according to their differences. This new generalized episode is a *specialization* of the parent generalized episode. The third possibility is that there is already a generalized episode indexed via this feature of the new case. In this situation, the new case is integrated into this substructure just as if it were the parent generalized episode. Unless there is some distinguishing feature, it is possible that no change will result and the new case will not be retrievable within this generalized episode. This yields a type of *forgetting* because the case cannot be differentiated from previous cases according to this feature. For more details see Kolodner (1984).

Because a problem solver's conceptual memory grows according to the sequence of problems encountered, it is possible that the particular sequence of cases cause the construction of either useless or incorrect generalizations. This is especially true when the problem solver has little or no knowledge to guide him in making generalization decisions. One solution to this problem is to monitor the usefulness of a generalization and remove generalizations that prove to be of no value (Kolodner, 1984). An alternate approach is to provide the problem solver with some domain knowledge in terms of a semantic generalization language for the particular domain (Mitchell, 1981). This knowledge can help the problem

solver avoid the construction of bad generalizations.

1.4.5 Generalized episodes in summary

Generalized episodes thus fulfill the three requirements we seek in organizing a conceptual memory. First, generalized episodes *organize knowledge* because they hold generalized information compiled from the cases they organize, and individual cases are indexed in these structures according to their differences from the norms for those concepts. Second, generalized episodes *allow retrieval* when two cases differ from the generalized episode in the same way. This is called a *reminding*, (Kolodner, 1984; Schank, 1982). Predictions based on the first case can then be used during case-based reasoning to analyze the new case (analogy). Generalized episodes correspond to domain components that are *similar* to each other, but need to be differentiated by *pertinent domain criteria*. Pertinent domain components of disputes include, for example, the dispute type, the disputants, their goals and arguments, and the disputed object. Third, generalized episodes *allow integration* when similarities between two cases are compiled to form a new memory schema with the structure just described (generalization). Over time, generalized case hierarchies are created (learning).

1.5 The MEDIATOR

The case-based approach to problem solving, including the memory organization and problem solving model introduced earlier, are implemented in a computer program called the MEDIATOR. The MEDIATOR is designed to provide advice about which mediation plans might be useful in the resolution of disputes encountered on a daily basis. This includes those encountered in taking care of children, in using and sharing objects, and in economic transactions. It also includes disputes encountered in reading the newspaper. As illustrated earlier, the knowledge we use day-to-day in resolving disputes can also be used to understand and predict the consequences of disputes we read or hear about.

The MEDIATOR has three major parts. First, its memory organization and indexing strategies allow previous similar cases to be recalled when appropriate. Second, it has rules for determining the most appropriate case when its memory returns more than one analogous past case. Third, its knowledge of where it is in the problem solving process serves as a guide in selecting those features of a past case which should be transferred to the current one. Its analogical processes help the MEDIATOR in classifying cases, choosing applicable mediation plans, predicting the results of a plan, and recovering from failures. The ability of the program to learn new cases and to resolve new disputes by the recall and transfer of information from previous cases gives evidence of the value of the case-based approach to problem solving.

In the case below, the MEDIATOR encounters the Sinai dispute, which was presented in section 1.2 above. In this example, case-based reasoning is used to aid in understanding the problem, in predicting solutions, in understanding the failed prediction, and in reinterpreting the case and selecting an alternate line of reasoning. Each use of case-based reasoning requires the location of previous cases, selection of the best one, and the transfer of appropriate knowledge to the new case.

Initially the MEDIATOR's memory holds information on four other cases: the Korean conflict, the Panama Canal dispute, a dispute between two boys fighting over a candy bar, and the sisters' dispute over an orange. User input is indicated by boldfaced type. The initial knowledge given the MEDIATOR about the Sinai dispute is that Egypt and Israel both want physical control of the Sinai, and that military means (arguments) have been used in previous achievement attempts.

I/O BEHAVIOR OF THE MEDIATOR FOR THE SINAI DISPUTE

```
(mediator sinai-dispute t)
Considering the following problem:
   Israel and Egypt both want the Sinai,
   which has been presented as ako M-DISPUTE.
(*DISPUTE*
 (PARTY-A (ISRAEL))
 (PARTY-B (EGYPT))
 (DISPUTED-OBJ SINAI)
 (ARGUMENT-A (*ARGUMENT* (ARGUER (ISRAEL))
                         (SUPPORT (*PHYS-CONTROL* (ACTOR ISRAEL)
                                                  (OBJECT SINAI)
                                                  (INST *MILITARY*)))))
 (ARGUMENT-B (*ARGUMENT* (ARGUER (EGYPT))
                         (SUPPORT (*PHYS-CONTROL* (ACTOR EGYPT)
                                                  (OBJECT SINAI)
                                                  (INST *MILITARY*))))))
```

In attempting to classify the dispute into one of its known dispute types, the MEDIATOR is reminded of two previous cases, the Panama Canal dispute and the Korean conflict. It chooses

the Korean conflict as most applicable since it shares more important features with the new dispute. It transfers the physical dispute classification as well as the goals of the recalled case from the Korean conflict case to the Sinai dispute. It then attempts to transfer the plan used successfully to resolve the Korean conflict by checking the plan's preconditions.

```
ATTEMPTING TO RECALL SIMILAR DISPUTES IN ORDER TO CLASSIFY THIS ONE...
   reminded of #<M-POL-DISPUTE 40306264> (Panama Canal was in dispute)
     because both disputants are of type M-POLITY.
   reminded of #<M-PHYS-DISPUTE 40306114> (Korea was in dispute)
     because both objects are of type M-LAND
         and both used M-MILITARY-FORCE to attempt *PHYS-CONTROL*
  Choosing #<M-PHYS-DISPUTE 40306114> (Korea was in dispute)

The current dispute will be referred to as #<M-PHYS-DISPUTE 40533552>
Attempting to transfer goal type from #<M-PHYS-DISPUTE 40306114>
ISRAEL and EGYPT are both inferred to have a M-PHYSICAL-CONTROL goal
  this is consistent with the normal uses of SINAI in this context.
Goal relationship is COMPETITION.

ATTEMPTING TO SELECT A MEDIATION PLAN
   TO RESOLVE THE DISPUTE IDENTIFIED AS #<M-PHYS-DISPUTE 40533552>.
Using previously recalled case, where Korea was in dispute.
   It was resolved using the plan known as "divide equally".
Checking for applicability of that plan to the current case..
I suggest that the plan called "divide equally" be used.
```

The MEDIATOR asks for feedback about its decision and is told both Egypt's and Israel's reactions. It attempts to come up with a new solution, and considers the failure of the suggested plan as the current problem to be resolved. It comes up with an explanation for the failure and a means of correcting it.

```
Is this a good solution? (Y or N) No.
   **** DIVIDE EQUALLY not acceptable ****

What happened?  (below is the English equivalent to the actual feedback)
 Israel said they wanted the Sinai to support national security.
 Egypt said they wanted the Sinai for national integrity.

ATTEMPTING TO EXPLAIN THIS FAILURE AND FIND A NEW SOLUTION.

ATTEMPTING TO RECALL SIMILAR FAILURES ...
   reminded of #<M-PHYS-DISPUTE 12475255>
    (two sisters quarrel over an orange)
      because in both the plan "divide equally" failed
          and both objects are of type M-PHYS-OBJ
Failure in that case was because of M-WRONG-GOAL-INFERENCE.
Transferring that classification to this failure.
Attempting to use previous remedy called
   "infer goal from resulting actions"
Unable to use previous remedy.
Considering other remedies useful for M-WRONG-GOAL-INFERENCE failures
Looking at the remedy called "infer goal from response"
Based on the feedback, I will replace ISRAEL's goal with
         a M-NATIONAL-SECURITY type goal and EGYPT's goal with
         a M-NATIONAL-INTEGRITY type goal.
Remediation complete.
```

The MEDIATOR next reprocesses the dispute. Because the problem has been interpreted previously, there is no need to reference previous cases until plan selection. The reminding process (left out this time) retrieves the same two cases as before. This time the additional information about the goals of the disputants causes the MEDIATOR to focus on a different exemplar, the Panama Canal dispute. Using that as a model, it suggests giving Egypt political control of the Sinai but giving military control to Israel.

```
Given this new information, I'll reconsider this problem.
Considering the following problem:
   Israel and Egypt both want the Sinai,
   which has been presented as ako M-PHYS-DISPUTE.
(*PHYS-DISPUTE*
 (PARTY-A (ISRAEL (*GOAL* (*NAT-SECURITY* (ACTOR ISRAEL)
                                          (OBJECT SINAI)))))
 (PARTY-B (EGYPT (*GOAL* (*NAT-INTEGRITY* (ACTOR EGYPT)
                                          (OBJECT SINAI)))))
 (DISPUTED-OBJ SINAI))
Goal relationship is CONCORDANT.
ATTEMPTING TO SELECT A MEDIATION PLAN
   TO RESOLVE THE DISPUTE IDENTIFIED AS #<M-PHYS-DISPUTE 40533552>.
ATTEMPTING TO RECALL SIMILAR DISPUTES ....

                        ...

There was one previous case with the same CONCORDANT goal relationship.
   It was resolved using the plan known as "divide into different parts"
Checking for applicability of that plan to the current case..
I suggest that the plan called "divide into different parts" be used.
INSTANTIATING EXPECTED CONTRACT RESULTING FROM PLAN APPLICATION.
  using the Panama Canal dispute to guide current contract construction
   matching ISRAEL with USA ...
   matching EGYPT with PANAMA...
   matching SINAI with PANAMA-CANAL...
   matching
(*GOAL* (*NAT-SECURITY* (ACTOR ISRAEL)
                        (OBJECT SINAI))) with
(*GOAL* (*MIL-CONTROL* (ACTOR USA)
                       (OBJECT PANAMA-CANAL)))...
   matching
(*GOAL* (*NAT-INTEGRITY* (ACTOR EGYPT)
                         (OBJECT SINAI))) with
(*GOAL* (*POLITICAL-CONTROL* (ACTOR  PANAMA)
                             (OBJECT PANAMA-CANAL)))...
   transferring other components of contract unchanged.
```

Figure 1-10

1.6 A guide to the reader

The remainder of this thesis is divided into four parts. The first part, chapter two, discusses the mediation task domain. It contains the necessary conceptual highlights of mediation as well as my representations of these concepts so that my examples in later chapters can be better understood. The second part of the thesis, chapters three through five, discuss the specifics of case-based reasoning within the three components of problem solving: understanding, planning, and failure recovery. The third part, chapter six, provides additional technical details of conceptual memory, my operational definition of similarity, and the heuristic selection of the most applicable case from a set of remindings. Chapter six covers those portions of memory processes not mentioned in this chapter. The fourth part, chapters seven and eight, provide a summary, conclusions, and compare this work to other research. In addition to a references section, two appendices are included to provide additional details of possible interest. Appendix A provides a complete collection of the cases used in developing the ideas embodied in the MEDIATOR. Appendix B provides two more examples of the program's behavior.

CHAPTER II

DISPUTE MEDIATION

2.1 Introduction

In this chapter, we discuss the important aspects of problem solving in the task domain of dispute mediation. The discussion will introduce the important components of dispute mediation and relate the problem solving process back to the abstract model of problem solving presented in chapter one. Along the way, we will discuss the mediator's role as a problem solver, the components of a particular type of problem known as a dispute, mediation plans available to resolve disputes, and contracts as a representation of dispute resolution. To begin, let us consider the following case:

CANDY DISPUTE-0

> A mother is on her way home from the library when she happens on two boys standing on a street corner quarreling over a candy bar. She overhears the first little boy shout, "I want it." The second boy responds, "So what, I want it too." Acting as mediator, the mother suggests that the boys divide the candy equally between them. Nodding their agreement, the boys split the candy and the mother continues homeward.

This case illustrates a type of planning and problem solving known as *third party mediation*. The mother in this case plays the role of the mediator. She is a third party to the dispute, and offers suggestions for dispute resolution. In general, in third party mediation, a non-involved problem solver, known as a *mediator*, helps resolve problems, or *disputes*, by suggesting possible *mediation plans* to the disputants for their acceptance. The candy dispute provides an example of a simple everyday dispute resolved by a common sense mediation plan which results in an implicit contract. In our problem solving analysis, our focus will be on third party mediation.

When the term mediation comes up, people often think of formal negotiations between big industries and labor unions. In reality, many common sense everyday situations involve mediation. For example, parents play mediator when they try to settle squabbles among their children. Family counseling services exist to try to help mediate problems between spouses and child discipline difficulties. Many couples turn to divorce mediation as an alternative to expensive and acrimonious divorce litigation. Realtors often act as mediators in bringing buyers and sellers together. Many executives are recognizing the importance of their mediation role in resolving labor grievances, breach of contract allegations, patent infringements, and internal management skirmishes (Main, 1983). Even judges play the role of mediator when they encourage litigants to settle out of court (Raiffa, 1982).

As a problem solver, a mediator has to *understand a given dispute, suggest plans for its resolution, verify that the results match his expectations*, and in case of violations, *figure out what went wrong and present a new plan*. These correspond to the stages of problem solving presented in chapter one. Because mediation often involves consideration of previous situations, we can transfer the case-based model of problem solving to this domain. The instantiation of that model in the dispute domain is illustrated below.

CASE-BASED DISPUTE MEDIATION

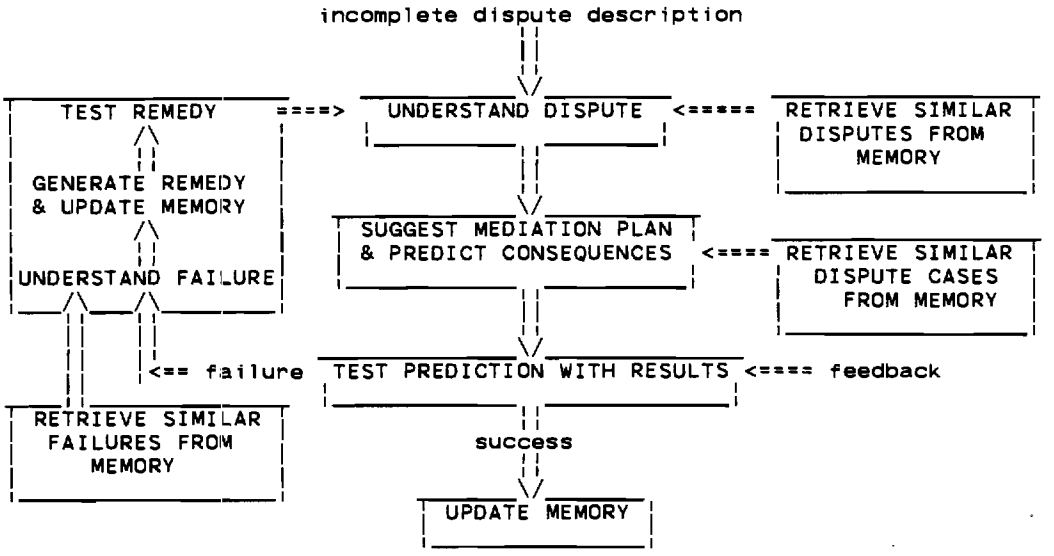


Figure 2-1

The notable differences between figure 2-1 and the more general model are in the specialization of the generic problem to its particular variant called a dispute. Thus, instead of a process for understanding the problem, we have a process for understanding disputes. Similarly, instead of a process for suggesting a generic resolution, we have a process for suggesting a mediation plan. In all other respects, the model shown in figure 2-1 is the same as that presented earlier. In the remainder of this chapter, we will look at important components of the dispute mediation domain.

2.2 Mediators

A famous mediator, William Simkin, in a semifacetious mood, once listed the following as the desirable qualities sought in a mediator (Simkin, 1971):

(1) the patience of Job
(2) the sincerity and bulldog characteristics of the English
(3) the wit of the Irish
(4) the physical endurance of the marathon runner
(5) the broken-field dodging abilities of a halfback
(6) the guile of Machiavelli
(7) the personality-probing skills of a good psychiatrist
(8) the confidence-retaining characteristic of a mute
(9) the hide of a rhinoceros
(10) the wisdom of Solomon

In a more reflective mood, he extended the list to include:

(11) demonstrated integrity and impartiality
(12) basic knowledge and belief in the collective bargaining process
(13) firm faith in voluntarism in contrast to dictation
(14) fundamental belief in human values and potentials, tempered by ability to assess personal weaknesses as well as strengths
(15) hard-nosed ability to analyze what is available in contrast to what might be desirable
(16) sufficient personal drive and ego, qualified by willingness to be self-effacing.

At least on qualities 1, 4, 8, and 9, an automated mediator would compare very favorably with Simkin's ideal mediator. The other qualities clearly require extensive knowledge and reasoning abilities.

The specific knowledge needed in any particular mediation situation will vary, but the basic underlying mediation roles remain the same. The first role of a mediator is to *propose reasonable solutions* to disputes. Sometimes the disputants are so close to a problem that they fail to consider solutions that a more objective party may notice. This is one advantage of third party mediation. In addition to their problem solving role, mediators perform a number of other important functions that support the settlement of disputes. For example, the mediator can make negotiations more effective by *collecting confidential material to see if a zone of agreement exists*. If private discussions indicate the existence of a possible agreement, then the mediator can focus the disputants toward this zone. The mediator can *keep negotiations going* when disputants refuse to negotiate directly with each other. In some acrimonious situations, this provides a face-saving means to hold communication channels open until the environment improves for further discussions. Occasionally, disputants will get hung up on a value or position that appears to provide no options for compromise. Under these circumstances, mediators can *help parties clarify their ultimate goals*. Finally, mediators can *deflate unreasonable claims* of disputants in order to overcome posturing or dirty tricks by disputants (Raiffa, 1982). Though we take into account each of the mediation tasks above, the research reported here focuses on the mediator's problem solving role: suggesting reasonable solutions.

Even though we have been guided in our conceptual analysis of the mediation task domain by examining the activities of experts, we by no means have attempted to model the reasoning of expert mediators. The formal mediation process has a very structured and constrained "protocol" for communication among the disputants and the mediator. There are "stereotypical" rituals of presenting "demands" and making "offers." Below this level, however, there is a basic problem solving process that we have attempted to model. We believe that many of the heuristics that professional mediators, like William Simkin, build up during a lifetime engaged in "protocol bound" dispute mediations can also be developed by any reasonably intelligent person during a lifetime of dealing in a common sense way with the domestic disputes that are a part of everyday life.

2.2.1 A mediator's objectives

Resolving a dispute requires two kinds of knowledge: (1) *domain knowledge* and (2) *planning knowledge*. By domain knowledge, we refer to knowledge about disputes, mediation plans, contracts, and specific details in the particular domain in which the dispute originates. For most disputes, the mediator selects a plan for resolving a dispute based on his knowledge of the disputants' goals and the disputed object's features.

Planners also need knowledge about their own planning objectives and policies. This is what we call planning knowledge (Stefik, 1981; Wilensky, 1983). Such knowledge, in mediation, includes the basic objective of the mediator, his policy when faced with making decisions in the absence of specific information, and the knowledge involved in assessing the "fairness" of a proposed mediation plan. These are general issues applicable to all dispute problems and are concerned with the problem solving process itself.

A mediator's basic objective is to resolve any given dispute in a way the disputants will find agreeable. As a third party, he cannot conclusively decide whether a particular mediation plan was a "good" solution unless the disputants provide him with the necessary feedback. He can, however, predict the reactions of the disputants to a suggested plan if he considers their goals. His resolution plans must therefore be chosen with the following basic premise in mind:

Mediation Basic Objective

To resolve disputes effectively, a mediator should suggest mediation plans that he believes the disputants will accept.

Figure 2-2

There are two implications to this. First, the mediator is discouraged from the random recommendation of mediation plans in a kind of blind search for solutions. Second, it encourages the use of previous cases. If a similar previous dispute employed a certain mediation plan successfully, then there is reason to believe that that same plan might be a reasonable solution approach for the current case.

Mediators, like most problem solvers, occasionally must make planning decisions for which there is incomplete domain knowledge. When faced with a dispute, such as the candy dispute, where the disputants' goals are in direct *competition* (i.e., both boys want the candy), the mediator has to make a basic decision whether to pursue compromise solution plans (e.g., divide the object between the disputants) or all-or-nothing type plans (e.g., give the object to one party). In the absence of specific information that can be used to direct this decision, the mediator is faced with making a guess. In mediation terms, such a decision is based on a *mediation policy*. In general, effective mediation requires the use of the following heuristic:

Mediation planning policy

Choose compromise plans before all-or-nothing plans for competitive disputes, unless it violates other mediation objectives.

Figure 2-3

Using this planning policy, the mother-mediator in the candy dispute would choose "divide equally" type compromise mediation plans when faced with competitive disputes over possession, unless other knowledge about the goals of the disputants were provided. If, for example, the boys were to explicitly tell her that they reject compromise solutions (e.g., "I want the whole candy bar!"), she would be obliged to consider "all or nothing" plans (e.g., "flipping a coin") which produce solutions in accord with the boys' goals.

Because mediators should demonstrate integrity and impartiality (this was quality number eleven in Simkin's list above), they must insure that their suggestions are perceived to be fair. There are two fairness doctrines to consider: *equality* and *equity* (Tedeschi and Rosenfeld, 1980). Equality is the mediation planning objective that insists that plans treat each disputant the same.

Mediation equality

Mediation compromise plans should treat the disputants the same, unless equity considerations are applicable.

Figure 2-4

Equality is achieved in the candy dispute by giving each boy an equal share.

Equity, on the other hand, is the mediation objective that insures that each disputant's share in an agreement reflects their contribution or ownership.

Mediation equity

Compromise mediation plans should insure that no disputant is allocated less of the disputed object than his proportion of ownership or contribution.

Figure 2-5

Since there was no question of ownership in the candy dispute (neither boy owned the candy), the equity objective was observed in conjunction with the equality objective when the candy was divided equally.

Sometimes, however, the two cannot be used conjunctively, as shown in the following variation of the candy dispute:

CANDY DISPUTE-1

A mother is on her way home from the library when she happens on two boys standing on a street corner quarreling over a candy bar. She overhears the first little boy shout, "I bought it, so it's mine." The second boy responds, "So what, if you don't give it to me, I'll flatten you!" The mother stops and says to the second boy, "If he owns the candy, he does not have to give it to you." After lecturing the second boy about fighting, she continues homeward.

In the candy dispute-1, the equity objective effectively preempts any attempt to divide the candy equally between the disputants by insisting that the owner should be awarded the entire candy bar. In general, equity is given precedence over equality.

2.2.2 An overview of mediation cases

In later chapters, we will demonstrate how previous cases can be used in making decisions during future problem solving efforts. To illustrate those points, we need to specify exactly what we mean by a case. A particular mediation case provides a record of the decision making that occurred during the stages of dispute resolution. It thus includes a description of the dispute and the mediation plan suggested for its resolution. The end product of a successful mediation decision is a contract produced by way of the mediation plan. This contract, too, is part of the case. Using this line of reasoning, we view a mediation case as the product of a staged mediation process. The following diagram illustrates this view.

AN ABSTRACT VIEW OF MEDIATION

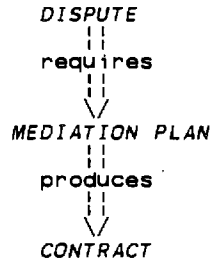


Figure 2-6

Each of these three stages provides important information about decisions made during the mediation process. The dispute description indicates decisions made concerning dispute classification and provides the specific dispute features that were observed or needed to be inferred. The mediation plan specifies the actions suggested in response to that specific problem, including decisions about what plan was selected and how it was refined. The contract indicates the expected final results of the mediation process, including the details of contract instantiation.

An especially important piece of additional intermediate information, for example, is the experience of recovery from failure. If we only recorded our final solution, we would not be able to recognize nor avoid previous failures. For this reason, a mediation case also includes those attempts at dispute mediation that failed. In general, a dispute may engender several mediation plans before one succeeds. Each of these attempts is recorded with the dispute for future consideration in mediation plan selection. The ideal mediation, reflected in the above diagram, corresponds to the final successful resolution in this situation.

In the remainder of this chapter, we will concentrate on the three major components of a mediation case: the dispute, the mediation plan, and the contract. First, the rationale for and representation of disputes is detailed in the next section. Following that, we will discuss mediation plans and their representation. Then we describe the content and representation of contracts. Finally in the last section, we will consolidate these components into a representation of a mediation case. In discussing these components, we will be identifying the primitive concepts (Schank, 1972) that are part of these mediation components and employing a frame representation technique (Minsky, 1975) to illustrate their interrelationship.

2.3 Disputes

Disputes can vary in terms of their disputants, the disputed object, the goals of the disputants, their arguments in support of their goals, and their setting. This section surveys the important features of disputes and presents representations for them. At the end, we will present both a classification scheme for disputes and a unified representation. As will become evident later, the features of disputes play an important role not only in problem solving but in memory organization and retrieval.

2.3.1 Disputants

Disputants are the parties engaged in the dispute. They are an absolute requirement for a dispute and are one of a dispute's most obvious components. Disputants can be *people* (e.g., the children in the candy dispute); *organizations* (e.g., the United Auto Workers); or *polities* (e.g., Israel and Egypt in the Sinai dispute). The disputants need not be of the same type, so all sorts of combinations are possible: people in dispute with organizations (e.g., professional athletes frequently get into salary disputes with their teams), people in dispute with polities (e.g., a home owner finds that the city has acquired his property by eminent domain), or organizations in dispute with polities (e.g., a company disputes its property tax levy from the city).

Disputants have important properties that support the understanding of disputes. For example, one very important feature of a disputant is his/her *goals*. Understanding a disputant's goals is essential to disputant resolution. Because it is so important, we will discuss disputant goals in a separate section. Another important set of features are a disputant's *themes* (Schank and Abelson, 1977). Themes are a source of inference for disputant's goals. Two types of themes are considered here: role and inter-personal themes. The first type of disputant theme is the *role theme*. For example, if we know that a disputant is a "merchant" then one possible goal inference is that he wants to engage in a "selling" action (i.e., ATRANS). This inference is derived from the "merchant" role theme. Another type of theme is the *interpersonal theme*. For example, if we know that two people are "married" then one possible inference is that they have the same goals, so that once the goals are learned for one they can be transferred to the spouse.

Some features are specific to particular types of disputants. For example, polities and organizations include a component designating the individual who acts as its leader. If we know the goals of the leader, then we can transfer their goals to their polity, or organization. As the above indicates, knowing what type of disputant is involved in a dispute enables a mediator to 1) infer missing information and 2) check that information transferred from other cases is consistent. For example, since Israel is a polity, one plausible inference is that it has a national security goal with respect to the Sinai. The same goal is not consistent, however, if we attempt to attribute it to a boy fighting over a candy bar.

Based on the above considerations, the representation of a disputant includes the disputant's name, goal, role, and interpersonal themes. The frame for BOY1, which was one of the disputants in the candy dispute is shown below as an illustration of how we represent disputants. We adopt the convention, in this and subsequent examples, of labeling mediation concepts with a "M-". For example, in the computer implementation, the symbol M-BOY is defined to be of type M-PERSON, which is used to represent the usual "ISA" inheritance relationship between two classes. Thus, M-BOY ISA M-PERSON and inherits slots for "name," "has-goal," etc. Other slots in the representation correspond to those components discussed above. Beside each slot is a brief comment describing the type of concepts that can fill the slot.

FRAME REPRESENTING "BOY1"

```
M-BOY isa M-PERSON
    name: BOY1    ; any string or atom identifier.
    has-goal: nil  ; a concept of type M-GOAL
    role-theme: nil  ; a concept of type M-ROLE-THEME
    inter-pers-theme: nil ; a concept of type M-INTER-PERS-THEME
    other-slots: nil  ; depends on the type of disputant
```

Figure 2-7

In general, disputes involve two disputants. There may, however, be more than two. When there are more than two disputants, the possibilities of coalitions being formed makes the identification of the sides of the dispute difficult. Often, common goals or thematic relationships, if known, can be used to identify coalitions that can then be viewed as a single party. For example, Ricky, Fred, Ethel, and Lucy are involved in a dispute over a vacation condominium. Instead of treating this as a four party dispute, if we notice that Ricky & Lucy as well as Fred & Ethel are related by the marriage theme, we can treat it as a two *couple* dispute, where each couple is viewed as a single conceptual disputant. This is illustrated in the frame representation below:

FRAME REPRESENTING "RICKY&LUCY" AS A COUPLE

```
M-COUPLE isa M-PERSON
    name: RICKY&LUCY
    has-goal: nil
    role-theme: nil
    inter-pers-theme: M-MARRIAGE isa M-INTER-PERS-THEME
                          husband: RICKY
                          wife: LUCY
```

Figure 2-8

This same approach allows us to reason in a common sense way even about disputes involving very large numbers of disputants. Consider the Law of the Seas Conference in which 160 nations were involved in a dispute over the unclaimed minerals in the sea beds of the world. Although there were many possible coalitions among these disputants, the disputants could be grouped into two coalitions based on their common goals. One coalition, made up of

114 developing countries, wanted to retain their stake in this mineral wealth even though they currently were incapable of exploiting it. The other coalition, made up of 46 developed countries, wanted to use their technology to begin mining operations for current sale or use. Using such groupings, it is possible to reason about some disputes involving many disputants as if they were two party disputes. All of the disputes considered in this research fall into the category of two party disputes.

2.3.2 Disputant goals

Disputant goals are an important part of both the disputants and the dispute. The key to resolving a dispute is the understanding of disputants' goals. Broadly speaking, we can classify disputant goals into one of three categories: (1) *physical goals*, (2) *economic goals*, or (3) *political goals*.

These categories permit us to infer potential actions on the part of disputants. Physical goals, in general, support predictions of the physical use of some object. For example, if the boys in the candy dispute have an INGEST goal, which is a physical goal, then we can expect that upon attaining part or all of the candy the boys will physically consume it. Economic goals allow inferences concerning the roles of the disputants as well as their possible actions. Normally, the existence of economic goals implies that one of the disputants will be playing the "buyer" role while another disputant will be the "seller." The expected actions in this situation involve an exchange of possession of some object for money. For example, when a vendor and a customer argue over the price of an object, the dispute derives from their conflicting economic goals. The buyer wants to establish the lowest settlement value, while the seller wants the highest possible value. Political goals permit expectations of actions directed toward the achievement of an abstract social state. For example, Panama wanted control of the Panama Canal returned from the United States to restore its national integrity. In this case, national integrity is the political goal of Panama that motivated its action to negotiate with the United States.

Goals can be instrumental to the achievement of other goals, as illustrated by the orange dispute. This leads to many possible goal arrangements. For example, physical goals can be instrumental to economic goals (e.g., physical control of an object enables the selling of the object). Physical goals can also be instrumental to political goals (e.g., occupation of a territory enables a polity to administer it or fulfill national ambitions, both political goals). Economic goals can be instrumental to both political and physical goals (e.g., with money one can buy an object (physically control) or acquire "favors", a political goal). Political goals can likewise be instrumental to either physical or economic goals (e.g., a polity can tax its citizens to achieve its economic aims and settle a territory to acquire physical possession). The disputants considered in this research are limited to a single goal of either the physical, economic, or political type.

Disputes arise because the disputants believe their goals to be in conflict. An individual disputant has to be able to determine when an action (or proposed action) either *supports* or *threatens* their goal attainment. For example, if the boys in the candy dispute want to eat the candy bar, then any action which gives them physical control over the candy "supports" their goal. Conversely, any action such as giving the candy to someone else "threatens" their goal. We model such reasoning by including among the components of the goal those components necessary to construct template sets of actions that represent *support sets* and *threat sets*.

We represent disputant goals as concepts of the type M-GOAL. These concepts contain components for the planner or individual who "owns" the goal, an expected action that is entailed by the goal primitive which provides the "header" portion for a constructed representation of the "desired state" of the planner, an "actor" component which often is the same as the planner, and an "object" component which is often the same as the disputed object. Other components of goals are the usual directional components (i.e., to and from), instrumentality components relating this goal to other goals, and a component indicating the modality of the action. All of these components are the basic pieces that are used to produce representations of the "desired state", the "support set" and the "threat set."*

---

*These representations are produced via procedural attachments. We demonstrate the representation of goals by showing below the M-INGEST frame that represents BOY1's desire to eat the candy in the candy dispute.

FRAME REPRESENTING BOY1'S INGEST GOAL

```
M-INGEST isa M-PHYSICAL-GOAL isa M-GOAL
     planner: BOY1
     actor: BOY1
     header: *ingest*
     object: CANDY1
     to: nil
     from: nil
     mode: nil
     inst: *physical-control*
     inst-to: nil
     desired-state: (*ingest* (actor BOY1)
                              (object CANDY1))
     support-set: (*physical-control* (actor BOY1)
                                      (object CANDY1)
     threat-set: ((*physical-control* (actor BOY1)
                                      (object CANDY1)
                                      (mode *NOT*))
                  (*physical-control* (actor (*VAR* &OTHER))
                                      (object CANDY1)))
```

Figure 2-9

2.3.3 Goal relationships

Because disputes have at least two disputants who have at least one goal each, we need to recognize the possible interactions between the disputants' goals. The interaction between the disputants' goals is termed the *goal relationship* (Wilensky, 1983).* Two goals can be *competitive* or *concordant*. We define a goal relationship as competitive when the achievement of one goal either prevents or impairs the achievement of the other. The candy dispute is a case that illustrates a competitive goal relationship. This is because neither boy can ingest the candy without preempting the other's ingestion goal. As one might expect, this is the prevalent goal relationship in disputes. We define a goal relationship to be concordant when the goals are not competitive. This includes those goals that are mutually supportive and those that have no interaction and thus do not interfere with each other. To illustrate the importance of identifying the proper goal relationship to dispute resolution, consider the following case:

ORANGE DISPUTE-0

> The mother arrives home from the library and finds her daughters quarreling over an orange. Recognizing the obvious similarity between this situation and her recent experience with the little boys, she suggests that her daughters stop quarreling and divide the orange equally between themselves. The girls agree to her suggestion. The first daughter peels her half orange and eats the fruit. But her sister peels her half, throws the fruit in the trash, and uses the peel to bake a cake.

As the orange dispute demonstrates, some concordant relationships are misinterpreted as competitive ones if the real goals of the disputants are not realized.

The goal relationship provides an important means of differentiating disputes, because it allows the mediator to make an initial characterization of the dispute as requiring compromise or not. For example, understanding the orange dispute as a competitive dispute under compromise planning policy means that the planner can directly focus on selection of an appropriate compromise plan without first considering "all or nothing" type plans. Likewise, if the mother had realized that her daughters' real goals were concordant, she could immediately have focused on plans that allow mutual goal achievement without having to consider compromise plans.

We derive the goal relationship of a dispute via a special procedure (see section 3.4.4) that examines the "threat sets" of each disputant's goal. If either of the disputant's goals threaten the other then the goal relationship is classified as "competitive". If the disputant's goals are not determined to be in a competitive relationship, they are classified as being "concordant."

---------- **We do not address the problems associated with tradeoffs among disputants who have multiple goals (see Raiffa (1982) for a discussion of these problems).

2.3.4 Disputants' arguments

Another important component of disputes is the *arguments* disputants advance in support of their goals. In third party mediation, the disputants' are assumed to be motivated toward a common agreement. Their arguments can usually be identified as *persuasive* arguments, because they are normally trying to persuade both the mediator and their opponent to accept their side of the issue.* Because of the persuasive nature of dispute arguments, we consider any action which attempts to advance one disputant's goals to be part of the broader argument concept. We thus consider actions such as "physical force" to be part of a disputant's arguments in addition to the things he says.

Arguments represent a source of information to support dispute understanding and selection of the best mediation plan. During understanding, for example, the arguments offer one source of inferring the disputants goals. But the argument is conceptually different than the disputant's goals. During planning, the information inferred from arguments helps direct plan selection decisions. For example, consider again Candy Dispute-1 presented earlier:

CANDY DISPUTE-1

> A mother is on her way home from the library when she happens on two boys standing on a street corner quarreling over a candy bar. She overhears the first little boy shout, "I bought it, so it's mine." To which the second boy responds, "So what, if you don't give it to me I'll flatten you!" The mother stops and says to the second boy, "If he owns the candy, he does not have to give it to you." After lecturing the second boy about fighting, she continues homeward.

The crucial difference between this version and Candy Dispute-0 is the new information available from the disputants' arguments. The mother reasoned that ownership of the candy, which was asserted in the first boy's argument, was sufficient justification to support an all or nothing resolution of the dispute, according to the equity principle.

This version of the candy dispute also provides examples of two types of persuasive arguments: *thematic arguments* and *dirty tricks*. The first boy's ownership assertion is an example of a thematic argument. Thematic arguments are defined as those based on common sense social conventions. The intuitive idea is that themes (e.g., ownership, parentage, marriage, etc.) account for certain goals and establish relationships between people or objects. In the case of possession disputes such as the candy dispute, establishing the ownership theme means that there exists an *owns* and *owned-by* relationship between the disputant and the object. Because it is a theme, ownership predicts that the owner will have a goal of possessing the owned object as well as the conventional knowledge that the owner usually should be awarded disputed objects. When the mother hears the ownership argument, she elaborates her conceptual representation of both the first disputant and the disputed object to reflect these inferences (e.g., boy1 owns candy1 and candy1 owned-by boy1).

The second boy's argument in Candy Dispute-1 is an example of a dirty trick. Dirty tricks are those arguments based on the actual or implied use of force or deception. The intuitive idea is that dirty tricks (e.g., threats, use of weapon, lying, etc.) are arguments used by disputants who lack more effective arguments (e.g., thematic or causal arguments).** This raises the problem of maintaining the integrity of the mediator's beliefs. Our solution to the use of dirty trick arguments by a disputant is to restrain the mediator from any inference based on such arguments if they are recognized. This protects the mediator, to a certain degree, from being misled. Thus when the mother recognized the second boy's threat against the first boy, she did not infer any special relationship between the second boy and the candy bar (as she did for the first boy).

Since disputants may lie, the possibility exists that the mediator will not recognize that the argument is a dirty trick. In that case, the mediator is likely to make incorrect inferences. For example, suppose the second boy had also argued that the candy should be his because he owned it. If the mediator simply adds the corresponding inferences as above, then we are faced with either concluding that there is joint ownership of the candy (a possible but incorrect conclusion here) or throwing out both inferences because of a constraint violation. The practical effect of either action is the same in this case. If the mediator throws out the inferences, the case is reduced to the original candy dispute, which leads to the "divide equally" solution. In the equal ownership case, the "divide equally" solution is still appropriate.

---

*This is contrasted with *adversary* arguments in which the participants do not expect to persuade their opposition (Flowers et al., 1982).

**Dirty tricks are negative specializations of the social act INVOKE (Schank and Carbonell, 1978; Carbonell, 1979).

We may never know for sure when a disputant is lying, but unless the mediator takes some action to discover the truth, the possibility exists that the the disputant who resorts to dirty tricks will be rewarded. For example, in the case where the second boy also claimed ownership he would be rewarded for his lying by getting half a candy bar that belonged to the first boy.

Principled mediators do not want to reward lying or other dirty tricks. So they often ask questions of the disputants to gauge the consistency of their argument. For example, the mother might take each boy aside and ask them where they bought the candy. She could then suggest that they all visit the store to verify the boys' stories. Mediators may also resort to deception in an attempt to evoke a differentiating response from disputants. For example, consider how effectively Solomon used deception in the case below to differentiate the real mother from the imposter.

BABY DISPUTE

> Two women came to Solomon both claiming to be the mother of a newborn baby and each claiming that the other wants to replace her accidentally killed child with the living one. There seemed to be no way to independently verify either woman's argument. Solomon said, "Divide the living child in two, and give half to the one, and half to the other." The real mother, fearing for the life of her child, begged Solomon to give the child to the second woman rather than kill it. The second woman agreed with Solomon's decision to divide the baby equally. Solomon, of course, gives the baby to the first woman.

Disputant arguments contain four key components: (1) the main point being argued for or supported, (2) the data or evidence used to support the main point, (3) the point being opposed by the arguer, and (4) the data or evidence used to attack the opposition's point. The four components are in evidence in the baby dispute above. Each woman argues symmetrically for her gaining physical control of the baby. Her support of this point is the thematic assertion of motherhood. Each, in turn, opposes the other woman's gaining physical control of the baby. The other woman is attacked by explaining that she had killed her baby and switched it for the living child. These four components of an argument are reflected in the representation of woman1's argument below. In this frame, the main point of the arguer is contained in the slot labeled "support-point." The slot labeled "support" contains the motherhood theme used to support the main point. By the same token, the point being opposed by the arguer is contained in the slot labelled "oppose-point." The information used to attack the "oppose-point" is contained in the slot labelled "attack."

A FRAME REPRESENTING "WOMAN1'S ARGUMENT"

```
M-PERSUASIVE-THEME isa M-ARGUMENT
    arguer: WOMAN1
    support-point: (*physical-control* (actor WOMAN1)
                                       (object BABY1))
    support: M-MOTHER isa M-INTER-PERS-THEME
                mother: WOMAN1
                child: BABY1
                expect-goal: (M-PROTECT M-CARE-FOR)
    oppose-point: (*physical-control* (actor WOMAN2)
                                      (object BABY1))
    attack: (*leadto* (ante (BABY2 (health -10)))
                      (conseq (*substitute* (actor woman2)
                                            (object BABY1)
                                            (for-obj BABY2))))
```

Figure 2-10

2.3.5 *Disputed objects*

All disputes involve a disputed object. As the above cases have illustrated, gout the disputed object can be just about anything, from a baby to a candy bar. A mediator needs knowledge about objects to infer plausible goals for the disputants and support the selection of an applicable mediation plan. There are three particular types of object knowledge that are basic to the mediator's reasoning about disputes: (1) *object function*, (2) *effect of object use*, and (3) *divisibility*.

Each type of knowledge provides the capability for a particular type of inference. Inferring a disputant's goal can be performed by reasoning about the function of an object. We have seen an example of this in Orange Dispute-0. The normal function of an orange is as the object in an ingest event. The mother used this knowledge to infer that the sisters wanted to eat the orange. The function of an object is context sensitive, however, as shown in the case below:

ORANGE DISPUTE-1

> The mother went to the fruit stand to buy some more oranges. A shopper at the fruit stand was quarreling with the manager over a particular orange. The shopper said it was half the size of the others and therefore should be half the price. The manager disagreed saying that the smaller ones were more flavorful which compensated for their size. The mother suggested that they split the difference. The manager and shopper agreed and everyone seemed pleased.

In understanding this case, the mediator should not infer that the disputants' both have the goal of ingesting the orange. While it is reasonable to assume that the shopper ultimately will consume the orange, the shopping context should restrict the goal inference for the manager. Within a shopping context, the normal function of an orange becomes that of the object in a buying and selling (an ATRANS) event and the role of the disputed object is filled by the price of the orange rather than the orange itself. In general, it is possible to use either the goal, if it is known, to infer a dispute type; or use the normal function of disputed objects within the known dispute type to infer the goal.

The effect of an object's use on the object constrains the selection of plans for resolution of a dispute. Some objects are *consumable* (e.g., candy and oranges) and may only be used once, while other objects are *non-consumable* (e.g. books and hammers) and can be re-used by different disputants at different times. This feature is important in the selection of mediation plans. For example, consider the following case:

BOOK DISPUTE-0

> Two students came to the librarian and asked to check out the same reserved book overnight. The librarian suggested that they take turns using the book. One check it out tonight, the other tomorrow night.

In this case, the disputed object will not be consumed if it is used by one of the disputants. Since the information in a book is not physically altered after its normal use, mediation plans such as "take turns" can be suggested.

Another feature of objects that influences the selection of mediation plans is the ability of an object to function after it has been divided. We will refer to objects which have this property as being *splittable*. In the orange dispute, the fact that the orange was splittable meant that the "divide equally" plan was a reasonable alternative. It is precisely because a book is not splittable that the same plan was ruled out for the book dispute. A half orange or half of a candy bar can still be used for their consumable purposes. Objects that are splittable also have associated with them the normal method that is used to accomplish their division. For example, liquids can usually be divided by pouring equal amounts into separate containers, while an orange is usually divided by using a cutting instrument such as a knife. Based on the examples presented thus far, splittable objects might appear to always be consumable objects. And conversely, non-consumable objects might be thought to always have the non-splittable feature. To illustrate that these features are independent, consider the following case:

AVOCADO DISPUTE

> The mother arrives home from the library and finds her daughters quarreling over an avocado. Recognizing the obvious similarity between this situation and her recent experience with the little boys, she suggests that her daughters stop quarreling and divide the avocado equally between themselves. The second sister protests that if the mother means to literally cut the avocado in two then the seed would be ruined.

The avocado seed is, like the rest of the fruit, a consumable object. Its normal function, however, is not ingestion but cultivation. In order to be a viable object for cultivation the seed must be whole. So the avocado seed is an example of an object which is consumable (i.e. cultivation alters the seed permanently) and not splittable (i.e. looses its functionality if split). It is this difference that keeps the second sister from accepting the "divide equally" plan in this case, in contrast to the original orange dispute.

Disputed objects are represented in terms of these primitive object feature types (e.g., M-CONSUMABLE-OBJ or M-FUNCTIONAL-OBJ). This approach has several advantages. First, it allows instance-level reasoning to be separated from class-level reasoning. For example, candy as a class is consumable, so we want to identify an instance of candy as a consumable object (i.e., candy1 isa consumable-obj). But any particular instance may fail to have what would normally be thought of as the defining feature of that class. For example, candy1 may be spoiled. Thus, candy1 may not be consumable even though candy in general is a consumable object. Second, object features can be classified into important groups that correspond to domain operators. The disputed-object preconditions for mediation plans, as we will see later, key on object feature types. For example, because candy1 is a splittable object, one of the preconditions of "divide equally" type mediation plans is satisfied. Finally, it allows new dispute objects to be defined consistently with previous experience, since they

were also defined using the same set of primitive feature types. This is important in insuring that new experiences are related properly to previous cases in memory.

These concepts are reflected in the representation of CANDY1 (the disputed object in Candy-Dispute-0), shown below. In this diagram, the primitive concepts are indicated as having an isa relationship to the "candy" concept. The slots inherited from the primitive object concepts are indicated by listing them below their corresponding concept.

FRAME REPRESENTING "CANDY1"

M-CANDY isa M-FOOD
name: CANDY1
number: 1
isa *M-CONSUMABLE-OBJ*
is-consumable: true
isa *M-SPLITTABLE-OBJ*
is-splittable: true
has-as-parts: *M-WRAPPER*
M-CANDY-PART
is-part-of: nil
isa *M-FUNCTIONAL-OBJ*
normal-usage: M-INGEST (in a physical context)
other-uses: nil

Figure 2-11

2.3.6 Dispute types

Despite the inherent variety of disputes, there are recurrent combinations of component features, which remain consistent during problem solving. These components are the goals of the disputants, the attributes of the disputed object, and the plans that can be employed to resolve the dispute successfully. These consistency constraints suggest the following broad dispute classes:

Physical disputes are conflicts between disputants over the possession or control of some object for ultimately a physical use.

Economic disputes are conflicts which revolve around the market value of some commodity.

Political disputes arise from conflicts over acceptable behavior among disputants pursuing political goals.

Each of these dispute classifications provide a context within which other inference can be directed. Once a dispute is classified, other inferences can be made in a consistent framework. For example, if a mediator decides that a dispute is a "physical dispute," then we expect the disputants to have "physical goals" which would be consistent with this hypothesis. Most of the cases presented earlier (e.g. the candy and book disputes) are examples of physical disputes. One example of an economic dispute (i.e., the dispute between the shopper and manager over the price of an orange) has previously been presented. Other examples of economic disputes are: (1) a customer and a vendor haggle over the price of an antique dish in a flea market, (2) a landlord and a tenant argue over the fair rental price of an apartment, and (3) the UAW (United Auto Workers) and Chrysler Corp. debate the union's wage requirements. In every case, an economic dispute involves a conceptual buyer and seller relationship between the disputants. This is an important differentiation in understanding because this initial decision will influence or color subsequent expectations of salience. In an economic dispute for example, we expect the disputed object to be the market value of an object and not the object itself.

The remaining class of disputes are called political because they involve one actor's attempts to influence another actor's behavior incidental to the achievement of a social or moral goal. Disputes in this category include the successful efforts of Panama to convince the US to return the Panama Canal, the jurisdictional disputes between unions like the UAW and the AFL/CIO which are scrambling to organize engineers and computer professionals (Sterling, 1982), and various talks between nations over the return of disputed lands (e.g., Great Britain and Argentina over the Faulklands or Egypt and Israel over the Sinai).

In the dispute domain, one dispute may be related to another dispute. This is sometimes referred to as *linkage*. Linkage means that the goals that are in conflict in this dispute are made instrumental to other goals, not necessarily involved with this dispute or disputant. For example, when a labor union negotiates a settlement with one company, the union must keep in mind that similar contract talks with other companies may use this agreement as an example. Thus their economic goals may become instrumental to their own political consistency goals. Without attempting to account for the full range of possible

dispute linkage, we will consider only the following relationships among dispute classes.

RELATIONSHIPS AMONG DISPUTE TYPES

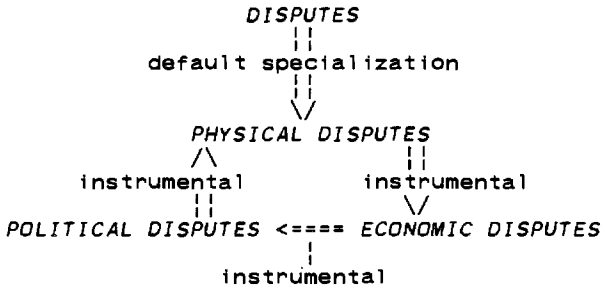


Figure 2-12

These relationships are a vital piece of knowledge that plays an important part in the default selection of dispute classes during understanding. The basic approach, which will be discussed in detail in a later chapter, is that when all else fails the physical dispute class is the default classification. If this classification is later found erronous, then the above relationship allows an orderly selection of the next best guess for dispute classification.

2.3.7 Representing disputes

In the preceding sections, we have described the important components of disputes: the disputants with their goals, goal relationship, and arguments; the disputed object and its characteristics; as well as the three major dispute classes. With this background, we are ready to present the overall dispute framework which is used to organize these separate pieces. The generic frame for a dispute (represented as a frame of type M-DISPUTE) has slots for those important components discussed above. The disputants are indicated as fillers for the "party-a" and "party-b" slots. Other slots for the arguments and disputed object are also evident. The illustration below shows the relationships of all these components and indicates some other pieces that will be discussed later.

GENERIC FRAME REPRESENTATION OF A DISPUTE

```
M-DISPUTE      e.g., physical disputes or economic disputes
    party-a: M-PARTY            e.g., a "person" or "polity"
                has-goal: M-GOAL          e.g., ingestion
                                planner: M-PARTY
                                actor: M-PARTY
                                header: a CD header  e.g., *ingest*
                                object: M-PHYS-OBJ
                                inst: instrumental goals
                                inst-to: supported goals
                                desired-state: an action in CD form
                                threat-set: list of CD actions
                                support-set: list of CD actions
                other-slots: depend on type of party
    argument-a: M-ARGUMENT  e.g., persuasive or adversarial
                    arguer: M-PARTY
                    sup-point: M-GOAL
                    support: some justification     e.g., M-THEME
                    opp-point: M-GOAL
                    attack: some justification      e.g., M-THEME
    disputed-obj: M-PHYS-OBJ
                      number: defaults to 1
                      other-slots: depend on the physical object
    setting: M-AREA or M-BUILDING
    party-b: M-PARTY, has the same structure as party-a
    argument-b: M-ARGUMENT, has the same structure as arg-a
    others: a list of M-PARTY
    usually-useful-plans: list of M-MEDIATION-PLAN
    other-plans: list of M-MEDIATION-PLAN
    enabled-mediations: list of M-MEDIATION
    instantiate-mediation: procedure to create a mediation frame
    specialize-dispute: procedure to transform representation
                        into one of the specialization classes
    goal-relationship: procedure to determine goal relationship
    rel-derivation: procedure to determine competition derivation
```

Figure 2-13

In addition to the dispute components mentioned above, the dispute frame includes procedural attachments used to represent the mediator's ability to do specific dispute related reasoning. These include procedures for creating mediation frames, determining the goal relationship of the dispute, and determining the source of competition if the dispute has a competitive goal relationship (Wilensky, 1983). For example, once a specialization class for the dispute has been determined, the procedure "specialize-dispute" knows how to transform the generic dispute representation into a more specific representation (e.g., physical disputes or economic disputes) as part of the understanding process. The slot labelled "usually-useful-plans" provides a list of known mediation plans the mediator can use as a source of potential actions to resolve the dispute. The slot labelled "enabled-mediations" provides a place holder where the mediator can record all his mediation attempts with respect to this dispute. To illustrate how an instantiated dispute is represented, the particular frame for the Candy-Dispute-0 (i.e., two boys are fighting over a candy bar) is shown below:

FRAME REPRESENTATION OF "CANDY DISPUTE-O"

```
M-DISPUTE  name: candy-dispute
    party-a: M-PERSON
                name: BOY1
                has-goal: nil
                role-theme: nil
                inter-pers-theme: nil
    argument-a: M-POSSESS
                    arguer: BOY1
                    sup-point: nil
                    support: M-PHYS-CONTROL
                                actor: BOY1
                                header: *phys-control*
                                object: CANDY1
                    opp-point: nil
                    attack: nil
    disputed-obj: M-CANDY
                    name: CANDY1
                    number: 1
                    isa: M-CONSUMABLE-OBJ
                             is-consumable: true
                    isa: M-SPLITTABLE-OBJ
                             is-splittable: true
                             has-as-parts: M-WRAPPER
                                           M-CANDY-PART
                             is-part-of: nil
                    isa: M-FUNCTIONAL-OBJ
                             normal-usage: depends on context
                             other-uses: nil
    party-b: M-PERSON
                name: BOY2
                has-goal: nil
                role-theme: nil
                inter-pers-theme: nil
    argument-b: M-POSSESS
                    arguer: BOY2
                    sup-point: nil
                    support: M-PHYS-CONTROL
                                actor: BOY2
                                header: *phys-control*
                                object: CANDY1
                    opp-point: nil
                    attack: nil
    usually-useful-plans: (M-DIVIDE-EQUALLY, M-TAKE-TURNS ...)
    enabled-mediations: nil
```

Figure 2-14

2.4 Mediation plans

Once a dispute has been understood, a plan is generated to resolve it. The generated plan, like the description of the dispute, is also an important part of the representation of a complete dispute mediation episode. In the domain of mediation, there are several "canned" plans available to use in resolving typical types of disputes. These canned plans called mediation plans, provide a means of structuring knowledge about the actions that can be taken to resolve disputes. Each mediation plan contains three important types of knowledge:

1. criteria for plan selection,
2. expected results of plan execution, and
3. an assessment of success or failure after plan execution.

Criteria for plan selection include plan use and conditions under which the plan is not recommended. Preconditions indicate the required state of the world before a given plan can be applied with some assurance of success. The notion of precondition is different here than in some other planners. Some planning systems (e.g., Sacerdoti, 1977) view preconditions as subgoals for the planner to achieve. Since a mediator cannot take actions to make a precondition true, a false precondition implies that the plan is not recommended for use in resolving the current dispute. Plan selection criteria can be provided explicitly or learned by experience. In the latter instance, the preconditions for a given plan will depend on the cases the mediator has previously attempted to resolve. Successful uses of a mediation plan provide the mediator with a set of features that describe the kinds of disputes the plan is appropriate for. Failed attempts to use a particular plan provide a set of features

describing those disputes for which the plan is not recommended.

The expected results of using a plan are also included as a part of the mediation plan. In the mediation domain, the expectations are contained in a "contract". We will discuss the specifics of mediation contracts in a later section. The point here is that the mediation plan contains these expectations and they are available for comparison to the actual results from plan execution.

Mediation plans also include follow-up assessments that record the judged success or failure of the problem solver after plan execution. A fully-instantiated mediation plan includes a record of the plan's successful and unsuccessful use. When a plan is instantiated for a particular dispute, it must be tailored to the specific details of that dispute. If the plan succeeds, that assessment will influence the integration of the mediation experience into memory, in this way defining what is learned. For example, whether or not a plan will be applied to a new dispute of this type depends on its assessment of success. When the plan fails, we need some mechanism to link this assessment and the dispute's subsequent resolution to this failure. This allows the problem solver to reason about previous failures and hopefully avoid them when this case is retrieved in the future.

We represent mediation plans using frames of the type M-MEDIATION-PLAN. The frame has component slots and procedures that support the planning concepts described above. For plan selection, the frame has a "precondition" procedure that evaluates the dispute in terms of its applicability criteria. For constructing expectations of plan execution results, the frame provides a procedure labelled "instantiate-contract." The constructed contract is then available in the "expected-contract" slot of the plan. For follow-up assessment after plan execution, the slot labelled "succeed" provides the basic assessment. The "failure-reason" slot in the mediation plan provides access to the subsequent error analysis and remediation experience. The actual plan that was ultimately used is available via the slot called "other-plan-that-succeeded" in the mediation plan. The organization of a generic mediation plan frame is illustrated below:

GENERIC FRAME REPRESENTING A MEDIATION PLAN

```
M-MEDIATION-PLAN                    e.g., divide equally, etc.
    precondition: procedure that tests for plan applicability
    expected-contract: M-CONTRACT    e.g., divided-obj-contract
    result: either "compromise" or "all-or-nothing"
    succeed: t or nil                ; from feedback evaluation
    instantiate-contract: procedure that creates the contract
                          that fills the expected-contract
                          slot above.
    other-plan-that-succeeded: M-MEDIATION-PLAN
                               filled only when plan has failed
                               and remediation succeeded.
    failure-reason: M-UNSUCCESSFUL-MEDIATION; e.g. bad inference
                        remedy: M-REMEDY
                        enabling-dispute: M-DISPUTE
                        results-confirmed: t or nil
                        feedback: list of M-EVENT
                        enabled-remediations:
                                 list of M-REMEDIATION
                        usually-useful-remedies:
                                 list of M-REMEDY
```

Figure 2-15

We have identified seven general plans that are useful for resolution of disputes:

General Mediation Plans

(1) Divide Equally
(2) Divide Unequally
(3) Take Turns
(4) Use Game of Chance
(5) Use Game of Skill
(6) Apply Recognized Standard
(7) Binding Arbitration

Figure 2-16

These general plans are defined briefly below and described in detail in the next sections.

Divide equally - the object is split into equal pieces, each party takes a piece.

Divide unequally - the object is divided either into functional subparts and each party takes the subpart associated with his goal, or according to the portion of ownership.

Take turns - each party alternates control or possession of the disputed object according to some prearranged schedule.

Use game of chance - the parties agree to resolve their dispute by the outcome of some random event.

Use game of skill - the parties agree to resolve their dispute by the outcome of some game or competitive sport.

Use recognized standard - in many dispute domains there are generally recognized standards that can be applied to indicate a resolution.

Binding arbitration - the parties agree to resolve their dispute in accordance with the decision of a respected third agent.

In the remainder of this section, we present specific details of the general mediation plans listed above and identify some of their better-known specializations. Preconditions for plan applicability and other differentiating information is presented with the description of each plan.

2.4.1 Mediation by equal division

Perhaps the most intuitive way of resolving a dispute is by equal division. This was the mediation plan suggested in Candy-Dispute-0 and has been the primary plan exemplar to this point. Divide equally, like all mediation plans, has preconditions that describe the situation where the plan is most applicable. There are four preconditions necessary for "divide equally" to be applicable:

1. the mediator has a compromise planning policy
2. the dispute has a competitive goal relationship
3. the disputed object is splittable and
4. the disputed object is not sharable.

The first precondition reflects the mediator's planning goal of effecting a compromise solution and implicitly confirms that the purpose of "divide equally" is to produce a compromise. The second precondition prevents "divide equally" from being selected for any concordant situations. The third and fourth preconditions insure that the disputed object has the proper attributes for division and that sharing is not a feasible alternative.

Dividing a disputed object equally among disputants requires instantiating a "division" action that will result in partitioning the disputed object into as many equal parts as there are disputants and an assignment of each part of the disputed object to each disputant. In the case of a disputed candy bar, the division can be accomplished by breaking or cutting. Candy-Dispute-0 did not specify how the boys assigned the pieces of the candy.

There are many possible "division" actions that can be used to instantiate a "divide equally" plan. The knowledge that an object is splittable normally includes the usual methods for accomplishing the split. Discrete physical objects usually have associated specific division actions. Continuous values, such as the economic worth of an object or the amount a window is open, is usually divided analytically. Thus if two disputants have different opinions on the subjective worth of some object (e.g., the orange in the orange dispute-1), then division is according to the normal notion of an average.

Assignment of the pieces of the partitioned object to each disputant can also be accomplished in several different ways. One way is to randomly assign each piece to a disputant (possibly using the "game of chance" mediation plan discussed below). Another method is to assign the pieces according to proximity, subjective value, or some other evaluative scheme. For example, when dividing a parcel of land among heirs, pieces can be assigned according to their distance from any land already owned by the heirs.

Because "divide equally" is applicable to a wide variety of disputes and has so many options for instantiation, several specializations are commonly employed. One specific version is "*one cuts, the other chooses*" (also referred to as "divide and choose" by Raiffa, 1982). It is primarily used for resolving physical disputes where the disputants distrust each other, such as Candy-Dispute-0. To further illustrate the applicability of the "one cuts" plan consider the following case:

SEA DISPUTE

During the "Law of the Seas" Conference, the issue of extracting mineral and other natural resources from the sea beds of the world effectively divided the conferees into the developed nations (those who were capable and anxious to extract these resourses) and the undeveloped nations (those who are currently unprepared to extract these resourses, but wanted to protect their future access and share of these non-renewable resources nonetheless). After much debate, the conferees agreed that the "non-territorial" waters of the world should be divided equally between the developed nations and the undeveloped nations. But this still left open the operational issue of which half of the sea beds should be assigned to which coalition. The undeveloped nations, without the technical knowledge to assess the relative value of different sea bed parcels, did not trust the developed nations to divide the sea beds fairly. Finally, it was agreed that when the developed nations are interested in a parcel of sea bed they should divide the parcel into two pieces and the undeveloped nations would choose which piece should be retained for themselves and the remaining piece assigned to the developed nations.

Another specific version of the divide equally plan is called "*split the difference*." This common sense plan is applicable in disputes where the disputed object can take on a continuum of values. Split the difference is most apparent in economic disputes, since price differences between the buyer and the seller can be eliminated via its application. To see how it can also be used in physical disputes, consider the following case:

WINDOW DISPUTE

Two men are quarreling in a library. One wants the window full open and the other wants it closed. Finally, the librarian suggests they split the difference and open the window half way.

The "divide equally" plans are summarized in the following diagram:

DIVIDE EQUALLY

General preconditions:

the mediator has a compromise planning policy
the dispute has a competitive goal relationship
the disputed object is splittable and
the disputed object is not sharable.

Results: Each disputant has equal share of disputed object.

Specialization: One cuts, the other chooses

Precondition: Disputants distrust each other.

Specialization: Split the difference

Precondition: Disputed object is continuous.

Figure 2-17

2.4.2 Mediation by unequal division

One of the advantages of mediation by equal division is that it is intuitively fair. But there are times when dividing something unequally is not only fair, but the preferred solution. A good mediation plan achieves as many of the goals of the disputants as possible. For this reason, some dispute situations are best resolved by an unequal division of the disputed object. Consider, for example, the following case:

FARM DISPUTE

Old MacDonald has decided to sell his farm in Georgia. The Thiele Kaolin Company, which extracts kaolin from strip mines, has learned that Old MacDonald's farm has a high kaolin potential. Thiele decides to buy old MacDonald's farm. But unbeknownst to Thiele, the Georgia-Pacific Company, a lumber concern, has also decided to buy Old MacDonald's farm as a source for current and future timber. Much to Old MacDonald's delight, a bidding war develops between the two companies. After several rounds of bidding have doubled the original asking price, Thiele and Georgia Pacific ask a realtor-mediator to help them resolve their dispute. The realtor mediator suggests that the companies divide Old MacDonald's farm into different parts. Georgia-Pacific buys the farm but without the mining rights. Thiele buys the mining rights. First, Georgia-Pacific will harvest any current lumber from the farm's surface. Thiele then gets to mine the deforested land for its kaolin, and then restores it for use as a tree farm by Georgia-Pacific.

This case is similar to the orange and avocado disputes presented earlier. The key point is that dividing the disputed object equally among the disputants would not have been the best possible (optimum) solution because an alternate division according to their real goals leads to better goal satisfaction.

Unequal division has two independent sets of preconditions corresponding to its two specializations. The first, "divide into different parts," applies to disputes with a concordant goal relationship where the disputed object can be split according to the goals of the disputants. This was the case in the farm dispute above. The other specialization of unequal division is known as "divide by equity." This plan applies to disputes where one or both of the disputants enjoy an ownership relation with the disputed object. The plan normally results in the disputed object being divided according to the percentage of ownership. When the disputants are equal shareholders, this plan is equivalent to "divide equally". To illustrate how this mediation plan works, consider the case below:

CONDO DISPUTE

Fred and Ethel wanted to be able to vacation in one of those fancy condominiums at the beach but couldn't afford to buy one. One evening while visiting their friends Ricky and Lucy, they mentioned their vacation dreams. Ricky suggested that the two couples form a partnership and buy a condominium to share. This seemed to be the ideal solution and both couples began working out the details. As it turned out, even in partnership with Ricky and Lucy, Fred and Ethel could only afford 25% of the purchase and maintenance costs of the condominium. So in the final arrangement, Ricky and Lucy paid 75% of all the costs. Later as the couples met with a realtor to sign the paperwork, Fred and Ethel began to draw up a schedule for the condo's use that allocated half the time to each couple. When Ricky and Lucy objected, the realtor suggested that a fair solution would be that Ricky and Lucy get to use the condo 75% of the time while Fred and Ethel use the remaining 25%. After realizing their error, Fred and Ethel apologized and began drawing up a new schedule.

The "divide unequally" plans are summarized in the following diagram:

DIVIDE UNEQUALLY

Preconditions: Object can be divided.

Results: Each disputant has different part or share of object.

Specialization: Divide into different parts

Preconditions:
The disputed object has different functional parts
The dispute has a concordant goal relationship

Specialization: Divide by equity

Preconditions:
One or both the disputants own all or a portion
of the disputed object.

Figure 2-18

2.4.3 Mediation by turn taking

Another intuitively fair mediation plan is the plan called "take turns". This plan specifies that the disputants perform some unspecified action in a prearranged sequential order. Thus the plan assumes that the action of one disputant does not prevent the other disputants from completing the sequence. The preconditions for this plan are tied to two general situations related to the specializations of "take turns." The first version of "take turns" is called "take turns using". It is applicable for disputes where the disputed object is non-consumable, as illustrated by Book Dispute-0 below:

BOOK DISPUTE-0

> Two students came to the librarian and asked to check out the same reserved book overnight. The librarian suggested that they take turns using the book. One check it out tonight, the other tomorrow night.

The next specialization of "take turns" is known as "take turns choosing." It is applicable for those disputes where there is a set of items to be distributed among the disputants. Usually the set of disputed objects is required to be equal or larger than the number of disputants. When the number of disputed objects equals the number of disputants and the disputed objects are all equivalent, "take turns choosing" is synonymous with "divide equally." The following case illustrates "take turns choosing":

BOOK DISPUTE-1

> Professors Boone and Crockett were both good friends and collectors of old books. One day they were walking to the university together, when they both spotted a few books strewn across the sidewalk in front of a small house. Boone picked up one of the volumes and was surprised to see that it was an eighteenth century printing of some Greek tragedies. Their interest aroused, the men soon discover that none of the books were printed later than 1914. About that time the door of the small house opened and a young man came out carrying another armload of books. Much to their delight, the young man gave all the books away. After calling a taxi and loading the books aboard, Boone and Crockett discuss how to divide up the books on the way back to their homes. The taxi driver over-hears the professors and suggests that they each take turns choosing a book until the books are all divided.

There are other specializations of "take turns" that key on the problem of deciding who is first in the sequence. One specialization which is applicable with both "take turns using" and "take turns choosing" is known as "worst goes first". Its precondition is some means of ordering the disputants. Once the order is determined, the most deserving or "worst" disputant is allowed to head the turn sequence or go first.* For example, consider the continuation of Book-Dispute-0 given below:

BOOK DISPUTE-0

> Two students came to the librarian and asked to check out the same reserved book overnight. The librarian suggested that they take turns using the book. One check it out tonight, the other tomorrow night. Which student should check it out first? This subproblem leads to a quarrel over who needs the book the most. The librarian asks each student for their grade point average (GPA). She suggests that the student with the lowest GPA go first.

---

*Another plan not discussed is "best goes first." This plan reflects the tradeoff involved in deciding that the worst is beyond hope as in triage situations, or in order to reward the better achiever for initiative and punish the worst for possibly lack of effort.

The "take turns" plans are summarized in the following diagram:

TAKE TURNS

General precondition:
The dispute has a competitive goal relationship.
One disputant's actions do not prevent another's action.

Results: Each disputant gets an equal portion or use of object.

Specialization: Take turns using

Preconditions:
The disputed object is not consumable and
The disputed object is not divisible.

Specialization: Take turns choosing

Preconditions:
There is a set of objects in dispute and
The number of disputed objects is equal to or greater than the number of disputants.

Specialization: Worst goes first

Precondition:
There exists some common characteristic which allows the disputants to be ordered.

Figure 2-19

2.4.4 Mediation by games of chance

When all else fails, disputes can be decided by using some random event. The prototypical "game of chance" is the flipping of a fair coin as officials do to decide matters at the beginning of a football game. In using this plan to mediate a dispute, the essential precondition is an agreement on the random event. This is especially important since all the disputants need to believe that their chances are no worst than the others in terms of subjective probabilities. The outcome from the application of this plan is unequal (i.e., one party wins the dispute), but each side has an equal opportunity.

Possibilities for a random event include rolling a die, selecting a card from a shuffled deck, drawing straws, drawing numbers (names) out of a hat, or guessing closest to a number concealed by a neutral party. In many stereotypical situations, especially in sports, there are specific "games of chance" that are part of the rules of play. We have already mentioned flipping the coin at the beginning of a football game. In tennis, the players spin a racket to determine who serves first. While in golf, the hitting order is determined initially by the spin/toss of a driving tee.

2.4.5 Mediation by games of skill

Rather than leaving their fate to a game of chance, some disputants prefer to resolve their differences by a game of skill. Often this plan is associated with disputes over "possession" of political or social status. The most dramatic example of this is the western shootout to determine who would have the status as the fastest draw. More mundane examples include typical sporting events (e.g., attaining gold medals) or any other measurable human skill. For example, lumbermen compete to determine the best lumberjack and sheep-shearers have competitions to determine who can obtain the most wool in a given time.

The precondition for this plan is that there must be some skill common to the disputants. If this is the case, then the possibility of resolving dispute by a game of skill exists. For example, consider this interesting case of mediation by a game of skill:

HORSE DISPUTE

Joe Bob and Billy Joe were sons of Big John, one of the most famous horsemen in South Texas. Big John owned not only the most horses, but the best horses. And the best of the best was Cass Ole. Both Joe Bob and Billy Joe dearly wanted Cass Ole for his very own. One day Big John overheard the two boys fighting over who deserved Cass Ole more. Both boys claimed to be the better rider. Big John told the boys that he would settle this argument by letting the boys run a horse race, the winner would get Cass Ole. To make things fair, Big John decided that each boy could choose from a corral of horses the steed the other was to ride.

The "games of skill" plans are summarized in the following diagram:

GAMES OF SKILL

General preconditions:
The dispute has a competitive goal relationship
and the disputants share some skill or capability.

Results: One disputant achieves his goal.

Specialization: Western shootout

Preconditions:
The disputants both have guns and
claim to be "the fastest draw."

Figure 2-20

2.4.6 Mediation by recognized standard

In many dispute domains, there are generally recognized standards for the resolution of disputes. These standards are sometimes codified in law or are otherwise generally agreed upon. Trivial examples of this are the games of chance or skill that have become part of the normal rules of play in some athletic contests, some of which were mentioned earlier. For example, in a pick-up game of basketball the standard means of choosing sides is to engage in a game of skill. The players each take turns shooting free throws, the first half of the players that make the shot are on one side.

In dividing a disputed object among competing agents, such as when an organization's budget must be distributed among disputing departments, the argument is often made that some disputants need (deserve) more of the disputed object (budget) than others. When these situations become politically or emotionally charged, the appeal to a standard allows hard decisions to be made. Consider the role that a traditional budget threshold standard plays in the following case from a United Press International (UPI) news story:

BUDGET DISPUTE

AM-JAPAN 0753
Japan Sets Limit on Military Spending
By CLYDE HABERMAN=
c. 1983 N.Y. Times News Service=

TOKYO _ The Japanese Cabinet put limits Tuesday on military spending for next year, setting in motion a fresh debate over whether the country gives its military too much or too little money.

Government officials anticipated complaints from the United States that Japan, despite planned increases, was still not providing enough funds for national defense. On the other side are domestic critics who feel that the military is getting more than its fair share at a time when the budgets of most government agencies are being slashed.

After long rounds of negotiations between the Defense Agency and the Finance Ministry, the Cabinet of Prime Minister Yasuhiro Nakasone compromised early Tuesday morning on a 6.88 percent limit on increases in military spending. It means that, at current exchange rates, the present military budget of $11.5 billion would rise to, at most, $12.3 billion in the 1984 fiscal year starting next April 1.

That 6.88 percent, however, is the ceiling; months will now be devoted to filling in specific details as to where the money should go. As is often the case, the figure ultimately approved could be smaller. In this year's budget, for example, the Cabinet originally established a limit of 7.3 percent, but that eventually was whittled down to 6.5 percent.

Among other things, Defense Agency officials worried that Tuesday's ceiling would set them back in plans for a sizable military buildup by 1988. They had been seeking an 8.9 percent increase. Originally, the Finance Ministry offered only 3.7 percent and, after negotiations, the two sides wound up splitting most of the difference.

One issue certain to arise is whether any increase stays within 1 percent of Japan's gross national product _ a threshold established nearly 10 years ago and one that has taken on an almost mystical significance here. Despite pressures from the United States, where military spending accounts for 6.6 percent of the GNP, no recent Japanese government has been politically prepared to go over 1 percent, and officials insisted Tuesday night that any increase in 1984 would also not pierce that barrier. upi 07-12-83 06:11 ped=

For a given dispute domain, a mediator needs to know what standards are possible so that they may be suggested at the opportune time. The problem is, of course, that there may be more than one way to apply a standard, or more than one possible standard that is applicable. For example, consider the following list of specialized standards (Fisher and Ury, 1981):

POSSIBLE MEDIATION STANDARDS

market value	relevant court decisions
precedent	moral criterion
costs	scientific judgement
tradition	professional ethics
efficiency	reciprocity

Precondition: Standard must be applicable to the dispute.
Results: Depend on standard applied.

Figure 2-21

Because different standards can be applied, there is always the possibility that the discussion over the appropriate standard will engender a subdispute. One party prefers to use one standard (which yields an advantage to them), while the other party suggests a different standard (which benefits their side). This subdispute may require mediation by a separate mediation plan. For example, use a "game of chance" to see whose standard will be applied; or "split the difference" between the results of applying the different standards. To illustrate

how disputes can arise of standards, consider this exchange between an insurance adjuster and his client (Fisher and Ury, 1981):

VALUE DISPUTE

Adjuster: Since the dump truck was totally at fault for destroying your parked car, we have decided that the policy applies. That means you are entitled to a settlement of $3,300.
Client: I see. How did you reach that figure?
Adjuster: That's how much we decided it was worth.
Client: I understand, but what standard did you use to determine that amount? Do you know where I can buy a comparable car for that much?
Adjuster: How much are you asking for?
Client: Whatever I'm entitled to under my policy. I found a secondhand car just about like it for $3,850. Adding sales and excise tax, it comes to about $4,000.
Adjuster: $4,000! That's too much!
Client: I'm not asking for $4,000 or $3,000 or $5,000, but for fair compensation. Do you agree it's only fair I get enough to replace my car?
Adjuster: OK, I'll offer you $3,500. That's the highest I can go. It's company policy.
Client: How does the company figure that?
Adjuster: Look, $3,500 is all you'll get. Take it or leave it.
Client: $3,500 may be fair. I don't know. I certainly understand your position if you are bound by company policy. Let me ask you to find out the basis for that policy. I'll call back tomorrow at eleven, after we both have a chance to study this matter.

****

Adjuster: OK, I've got an ad here in today's paper offering a '78 Fiesta for $3,400.
Client: I see. What does it say about milage?
Adjuster: 49,000. Why?
Client: Because mine only had 25,000 miles. How many dollars does that increase the worth in your book?
Adjuster: Let's see...$150.
Client: Assuming the $3,400 as one base, that brings the figure to $3,550. Does the ad say anything about a radio?
Adjuster: No.
Client: How much extra for that in your book?
Adjuster: $125.
Client: How much for air conditioning?

****

Later that day the client picked up a check for $4,012 from the insurance adjuster.

2.4.7 Binding arbitration

While some disputants prefer the risk of a resolution by way of a game of chance, others prefer having someone else play a more direct role. Technically speaking, a mediator's role is as an aid to the disputants. He can help, but he does not have the authority to dictate a solution. Occasionally, disputants will ask a mediator to settle the dispute for them. It is at this point that the mediator technically becomes an *arbiter*. An arbiter, like a mediator, is interested in fairness, but after he has determined the facts and heard the arguments, he has the additional authority to impose a resolution. For all practical purposes, an arbiter is judge and jury. Whether a mediator is asked to arbitrate at the request of the disputants or not, a mediator can always suggest that the disputants submit their dispute to binding arbitration as a means of resolution. Preconditions to this mediation plan are that the disputants have to agree to it and no other solutions can be found.

There are two types of arbitration in technical terms, *conventional* and *final-offer.** Conventional arbitration is exemplified by the discussion above and is structured very much as normal litigation. There is a wealth of case precedents and previous decision guidelines available to the arbiter, analogous to the case law available to lawyers and judges. Contract disputes are typical of the disputes that are heard in arbitration. In many states,for example, public-service employees cannot strike for higher wages so their demands and grievances are subject to binding arbitration.

The other type of arbitration is called final-offer arbitration. The procedural details are as follows: the disputants bargain directly with or without the aid of a mediator. If they come to a point where no further progress seems possible, the disputants each seal their final offer and give it to the arbiter. The arbiter alone then chooses between the final offers. There is no compromise and the selected offer becomes binding on all parties. Professional baseball, for example, uses this method to resolve player salary disputes.

---

*This distinction is purely for formal arbitration cases not for common sense arbitration.

The "binding arbitration" plans are summarized in the following diagram:

BINDING ARBITRATION

General precondition:
The dispute has a competitive goal relationship.
The disputants agree to let a third party make a binding decision.

Results: Depends on the type of arbitration.

Specialization: Conventional arbitration

Preconditions:
The disputants agree on the arbiter.
Results:
The arbiter can decide on any "fair" settlement.

Specialization: Final-offer arbitration

Precondition:
The disputants agree on the arbiter.
Results:
The arbiter rules in favor of one disputant.

Figure 2-22

2.4.8 Mediation plans in summary

In terms of their results, two of these general plans (divide equally and take turns) produce compromise solutions. Three produce all or nothing type solutions (divide into different parts, use game of chance, and use game of skill). And two can produce either (apply recognized standard and binding arbitration). This knowledge, in conjunction with the goal relationship classification of the dispute, allows a dispute mediator to quickly eliminate plans that do not match the current planning policy, rather than having to consider all plans.

Each general mediation plan is applicable to all types of disputes. For example, divide into different parts was employed to resolve Orange-Dispute-0, which was a physical dispute, as well as the Panama Canal dispute, which was a political dispute, and the farm dispute, which is an economic dispute. This means that the mediator has to do more complex reasoning than just looking at the dispute type in selecting the most appropriate plan, even when the choices have been narrowed. By a process of elimination, the preconditions for each plan could be tested to eventually select one. But, as we will see later, the recall of a previous similar case often allows us to by-pass this step and select a plan more directly.

2.5 Mediation contracts

The last component of mediation experience is the expected results of applying the chosen plan. We represent the expected results of a plan in a contract. A *contract* is the normal product resulting from the application of a mediation plan to a dispute. It represents a solution to the dispute problem. Contracts package mutual *expectations* concerning the cooperative actions of the parties involved. See Dyer (1983) for more details of conceptual contract components. Because contracts are derived from the dispute, they contain information about the disputants and the disputed object. If the mediation plan resulted in a compromise, then the original goals of the disputants are reflected in the contract as a *partial goal satisfaction*. This is sometimes reflected in the disputants being allocated a portion of the disputed object. Combining the disputants' original goals and the results of the mediation plan leads to an expectation concerning the actions of the disputants subsequent to the mediation experience.

For a given dispute, a particular mediation plan produces a specific type of contract. There is a different contract type for each mediation plan. In Candy-Dispute-0, for example, a resolution via the "one cuts, the other chooses" plan means that an instance of a "divided object contract" is created. This particular contract requires that one boy play the role of the "cutter" responsible for dividing the disputed object. The other boy plays the "chooser," responsible for selecting first from among the pieces. In much the same way that a script (Cullingford, 1981; Schank and Abelson, 1977) allows expectation-based understanding, the normal sequence of events in the instantiated contract serve two purposes in planning and follow-up. First, during plan selection, they allow the consequences of a given mediation plan to be *simulated* and judged in comparison to other alternatives. This appraisal is normally deeper and subordinate to the precondition testing that is part of initial plan consideration. Second, when feedback about the suggested plan or implementation of the plan is evaluated, expectations provide a means of confirming success or recognizing failure. This will be discussed below.

2.5.1 The role of contracts in assessing results

Contracts play an important role in assessing feedback to determine the results of a mediation attempt. They package predictions in support of feedback evaluation.

Sometimes the contract that resolves a dispute is implicit, as is the case in most common sense disputes like Candy-Dispute-0. Other times, a contract is the explicit form of an agreement. For example, when labor unions and company management settle their differences, their agreement is formalized by the terms and conditions of a contract. When countries settle their disagreements, they sign a contract called a treaty which specifies the details of their pact. All contracts, whether implicit or explicit, package mutual *expectations* concerning the cooperative actions of the parties involved. For Candy-Dispute-0, this means that the boys realized that by agreeing to the "divide equally" mediation plan, they had created a "divided object contract". This contract implies that one of the boys is to divide the candy into two equal pieces and give one piece to the other boy.

Expectations allow the problem solver to test that results of applying the plan match predictions. For the disputants, these expectations allow them to *monitor* each other for compliance during the life of the contract. For those external to the dispute, expectations from the contract permit verification of dispute resolution. As an example of execution monitoring, we can imagine the first boy tearing off a small piece of the candy and offering it to the second boy. A renewal of the dispute is likely in this case. To illustrate the verification process, recall that the mother in Candy-Dispute-0 is external to the problem. She can decide that she has suggested a plan that actually resolves the dispute by analyzing the boys' actions, and comparing them to what she expected. This evaluation determines whether or not the mother views this as a *successful dispute resolution.*

A mediator can never be sure that a recommended solution is a good one until the feedback from both the disputants and the actual events confirm the contract expectations. Thus the mediator is faced with a situation where he has a very limited ability to evaluate solutions.

This interesting characteristic of dispute mediation means that mediators are forced into a sort of generate and test problem solving mode with two separate evaluation points. One is an evaluation of planning options. The other is an evaluation of plan execution. First the mediator must evaluate possible mediation plans and select the most appropriate. Feedback from the disputants tells the mediator that the plan is *acceptable*. For most AI planning systems, this would represent the only form of evaluation. If the mediator is to learn from execution failures, then there must be another evaluation when feedback from plan execution is available for comparison to expectations. It is this second round of evaluation that permits the mediator, or any problem solver, to assess his success in resolving problems. For example in the orange dispute, because her expectations were not fulfilled and there existed a "better" mediation plan, the mother should evaluate the original "divide equally" plan as a failure and indicate that the dispute should have been resolved using the "divide into different parts" plan. Without this second round of evaluation a mediator would never learn from failures like the orange dispute.

2.5.2 Representing contracts

This section presents those components we require to be represented in conceptual contracts. Because the contract is a derivative of the dispute via a mediating process, the contract inherits such features as the contract parties and the disputed object from the dispute itself. Of these inherited features, the disputed object normally has been or is projected to undergo some transformation as a result of the mediation plan. For example, if the mediation plan calls for the disputed object to be divided, then there will be parts of the object that need to be instantiated as separate objects and the original disputed object marked as no longer in existence.*

The most important component of the contract is the events predicted to occur as a consequence of the contract. In the book dispute example, the student with the lowest GPA will first check the book out of the library, read it, and then return it the next day. Next the other student will check it out, read and return the book. These expectations are important because they permit the disputants and mediator to recognize violations when *feedback* does not match these predictions. Expectation violations trigger error recovery actions as explained above.

In the limited sense they are used in this research, the concept of a contract is represented by the M-CONTRACT frame. There are frames associated with each specialized type of contract. For example, a frame of type M-DIVIDED-OBJECT-CONTRACT is shown in Figure 2-23.

GENERIC FRAME REPRESENTING "M-DIVIDED-OBJECT-CONTRACT"

```
M-DIVIDED-OBJECT-CONTRACT isa M-CONTRACT
    disputed-obj: M-PHYS-OBJ  from the dispute, e.g., candy1
    part-a: M-PHYS-OBJ        e.g., half of candy1
    part-b: M-PHYS-OBJ        e.g., other half of candy1
    party-a: M-PARTY          from the dispute
    party-b: M-PARTY          from the dispute
    duration: symbol indicating expected contract life.
    mediation-plan-used: M-MEDIATION-PLAN
    predictions: M-POSSIBLE-EVENTS ; the expected actions if
                    results-ok: events expected from chosen plan
                    misunderstanding: events that can mean a
                                      misunderstanding error.
                    context: events that can mean the context
                             has been inferred incorrectly.
                    policy: events that can mean a policy
                            inference is in error.
```

Figure 2-23

---

*The object representation cannot simply be "deleted" because later we will want to be able to refer to the object "the way it was."

As can be seen in Figure 2-23, the disputants and disputed object have been carried over into the contract frame. Other than these inherited features of the dispute, the contract frame does not directly refer to the originating dispute representation. The contract does, however, refer back to the mediation plan instance that produced the contract via the "mediation-plan-used" slot. But, there is no direct path back to the mediation experience or dispute representation from the mediation plan representation. This means that a backward chaining reasoning process could not be employed with the current knowledge structures. Parts of the disputed object assigned to the disputants during mediation are indicated by the "part-a" and "part-b" slots. The expectations of disputant actions as a result of the mediation are located via the slot labelled "predictions." This slot is filled by objects of the type M-POSSIBLE-EVENTS. Possible events are explicit actions that are either as expected (i.e., "results-ok") or are one of several known types of failures. These are indicated in the M-POSSIBLE-EVENTS frame as "misunderstanding" errors, "context" errors, or "policy" errors. We will discuss the specific details of failure recovery in chapter five.

To illustrate how error recovery is triggered by contract expectation violation, consider the "divided object contract" instance shown below. It was the result of the application of the plan "one cuts, the other chooses" in Orange Dispute-0. This one shows the expectations that are used to trigger failure recovery in that case. This process will be demonstrated in chapter five.

FRAME REPRESENTING AN INSTANCE OF THE "M-DIVIDED-OBJECT-CONTRACT"

```
M-DIVIDED-OBJECT-CONTRACT isa M-CONTRACT
    disputed-obj: ORANGE1
    part-a: M-DIVIDED-OBJ
                name: sister1's half
                was-part-of: ORANGE1
                portion: *half*
    part-b: M-DIVIDED-OBJ
                name: sister2's half
                was-part-of: ORANGE1
                portion: *half*
    party-a: SISTER1
    party-b: SISTER2
    duration: *orderminutes*
    mediation-plan-used: M-ONE-CUTS-OTHER-CHOOSES
    predictions: M-POSSIBLE-EVENTS
                    results-ok:
                       ((*ingest* (actor SISTER1)
                                  (object "sister1's half))
                        (*ingest* (actor SISTER2)
                                  (object "sister2's half")))
                    misunderstanding:
                       ((*ingest* (actor SISTER1)
                                  (object "sister1's half))
                                  (mode *not*))
                        (*ingest* (actor SISTER2)
                                  (object "sister2's half")))
                                  (mode *not*))
                    context: nil
                    policy: nil
```

Figure 2-24

In this case, the "divided object" contract predicts, in conjunction with the girls' understood ingest goals, that each sister would eat her half of the orange. Thus when the mother learns, via feedback, that the second girl has used the peel from her half to bake a cake, her expectations are violated. In this case, the violation points out an error in the mother's understanding of the dispute. If the mother is to learn anything from this failure, she should introspectively resolve the dispute problem using the sisters' real goals as indicated by their later actions. Such an introspection, in this instance, should lead to the use of the "divide into different parts" plan to produce a contract whose predictions match the actual results.

2.6 Representation of mediation experiences

In general, a problem solving experience consists of a problem, a plan to resolve the problem, and a record of the solution usually in terms of the expected versus the actual results. In general, this experience is remembered as either a success or a failure based on a subjective evaluation of the results. We can specialize these ideas for the mediation domain as follows: disputes are a type of problem, mediation plans are known solution plans, and contracts are the record of the expected and actual solution results. The evaluation of this experience results in a determination of either a successful or unsuccessful mediation. In the past three sections, we have presented the three major portions of dispute mediation

experiences: the disputes, mediation plans, and contracts. We can package these separate components into a complete mediation experience as illustrated in Figure 2-25.

GENERIC FRAME REPRESENTING A MEDIATION CASE

```
M-MEDIATION          e.g., a successful or unsuccessful mediation.
    dispute: M-DISPUTE
    mediation-plan: M-MEDIATION-PLAN      e.g., divide equally
                        expected-contract: M-CONTRACT
    predictions: M-POSSIBLE-EVENTS    ; from contract
                      results-ok: list of expected actions
    results-confirmed: t or nil              ; from feedback.
    feedback: list of M-EVENT ; e.g., observed or reported acts
    remediations: list of M-REMEDIATIONS
    usually-useful-remedies: list of M-REMEDY
    specialize-mediation: a procedure that transforms mediation
                          after evaluation into either a success
                          or failure representation.
```

Figure 2-25

As can be seen above, a mediation experience contains several other components in addition to the three described. In particular, there are components concerned with feedback evaluation and failure recovery. The representation above includes a "feedback" slot to record the actual events that occurred after mediation. When a mediator evaluates the feedback, he decides whether the mediation was a success or not. This is recorded two ways in the above frame. First, if actual feedback has been received and evaluated, then the "results-confirmed" slot is used to indicate a boolean success or failure. At the same time, the procedure "specialize-mediation" is invoked to reformulate the mediation case into a frame of either M-SUCCESSFUL-MEDIATION or M-UNSUCCESSFUL-MEDIATION type. If the case was an unsuccessful one, the slot labelled "usually-useful-remedies" provides a set of possible remedies for the error. And, the slot labelled "remediations" records the results of error recovery. The error recovery aspects of a mediation case are explained further in chapter five.

In the diagram below, we will illustrate the instantiation of a M-MEDIATION type frame. This particular frame represents Orange Dispute-0 at the point where the mother-mediator has learned, through feedback, that her daughter has used the peel (i.e., PEEL1) from orange1 to bake a cake (i.e., CAKE1). She has concluded that her mediation was a failure and the frame is so labelled. No error recovery actions are yet indicated.

INITIAL FAILURE OF THE "ORANGE DISPUTE-0" MEDIATION CASE

```
M-UNSUCCESSFUL-MEDIATION isa M-MEDIATION
    dispute: orange-dispute
    mediation-plan: M-ONE-CUTS-OTHER-CHOOSES
                        expected-contract: M-DIVIDED-OBJ-CONTRACT
    predictions: M-POSSIBLE-EVENTS
                    results-ok:
                        ((*ingest* (actor SISTER1)
                                   (object "sister1's half))
                         (*ingest* (actor SISTER2)
                                   (object "sister2's half))
    results-confirmed: nil
    feedback: ((*ingest* (actor SISTER1)
                         (object "sister1's half))
               (*prepare* (actor SISTER2)
                          (object CAKE1)
                          (inst (*physical-control*
                                    (actor SISTER2)
                                    (object PEEL1)))))
    remediations: (M-REMEDIATION
                      name: failure of orange dispute
                      ...)
    usually-useful-remedies: (M-USE-ACTUAL-EVENTS ...)
```

Figure 2-26

2.7 *Summary*

In this chapter, we presented the dispute mediation task domain. In order to illustrate case-based reasoning in this particular domain, we needed to identify the appropriate problem solving components of the domain. First, the mediator is specified as the problem solver. A mediator is a non-involved third party that helps to resolve disputes by suggesting possible solutions. To aid in doing this task, the mediator stores previous cases in memory to use in later reasoning. A mediation case contains four primary components: the dispute (a problem), the mediation plan (possible action by the problem solver), a contract (the solution), and results evaluation (feedback evaluation and failure recovery as necessary).

A dispute has many features that must be recognized and represented. These include: the disputants, the disputants' goals, the disputants' goal relationship, the disputants' arguments in support of their goals, and the disputed object. Representations for all these components were presented.

We have identified seven general classes of mediation plans that represent canned actions useful for resolving certain types of disputes: equal division, unequal division, turn taking, games of chance, games of skill, use of a standard, and binding arbitration. Each of these general classes have specializations that address specific stereotypical situations.

The mediation contract represents the solution to a dispute. The contract is produced by applying a chosen mediation plan to the specific dispute. Our major interest in the contract is in its role as a holder of expectations. These expectations are then available for later evaluation and follow-up.

CHAPTER III

CASE-BASED REASONING IN PROBLEM UNDERSTANDING

"*First*. You have to *understand* the problem." (Polya, 1945)

3.1 Introduction

BABY DISPUTE

> Two women came to Solomon both claiming to be the mother of a newborn baby. Each woman accuses the other of stealing her child as a replacement for the others' child which had been accidentally killed. There seemed to be no way to independently verify either woman's argument. Solomon said, "Divide the living child in two, and give half to the one, and half to the other." The real mother, fearing for the life of her child, begged Solomon to give the child to the second woman rather than kill it. The second woman agreed with Solomon's decision to divide the baby equally. Solomon, of course, gives the baby to the first woman.

Would that we all had the wisdom of Solomon to accurately understand problems. Solomon clearly understood the different goals that motivated the two women. A real mother would be motivated by the natural desire to protect and nurture her child, while a woman who had accidently killed her child would desperately grasp at anything to avoid the shame attendant with such an admission. With this understanding, Solomon masterfully devised a plan to evoke a differentiating response which would allow him to identify the real mother.

The focus of this chapter is on understanding problems. Solomon's dilemma illustrates some of the difficulties faced by mediators in particular and problem solvers in general. Key components of problems, such as the goals of disputants, are not always obvious. We must infer goals and other unspecified details of the problem from the information provided. In addition, erroneous information must be recognized and taken into account. In the example above, for instance, people have no difficulty realizing that one of the women is lying. How can we incorporate these same consistency checks into an overall understanding process? What exactly is involved in understanding a problem? These are some of the questions discussed in the following sections.

While our primary purpose in this chapter is to investigate the use of previous cases in understanding problems, we cannot ignore the relationship of this process to the entire problem understanding process. For this reason, we will first present our view of problem understanding. This provides the perspective and context for later sections which concentrate on the specific case-based reasoning processes employed at different points in the problem understanding task.

At all times, our goal is to present case-based reasoning in an integrated perspective with other reasoning processes. We do not see case-based reasoning as a replacement for other methods of problem understanding, but as a heuristic enhancement for what would otherwise be a static process. In later sections, when we present specific case-based algorithms, their general form will be: first, attempt to make the decision by reasoning analogically from a recalled case, then use normal default reasoning to make the decision if analogical transfer is not appropriate. We believe powerful problem solvers need multiple lines of reasoning. Our model of problem solving using case-based reasoning offers one method of integrating multiple lines of reasoning.

3.2 An overview of problem understanding

Problem understanding has long been recognized as the first, if not most important, stage of problem solving (e.g., Bobrow, 1968; Greeno, 1977; Hayes and Simon, 1979; Paige and Simon, 1979; Polya, 1945). In general, problem understanding is a process that receives an initial, often incomplete, problem description from the environment and constructs an internal representation of the problem. This problem representation is then available for use during further problem solving. The place of problem understanding in our model is highlighted in our overall process model, originally presented in chapter one and repeated below:

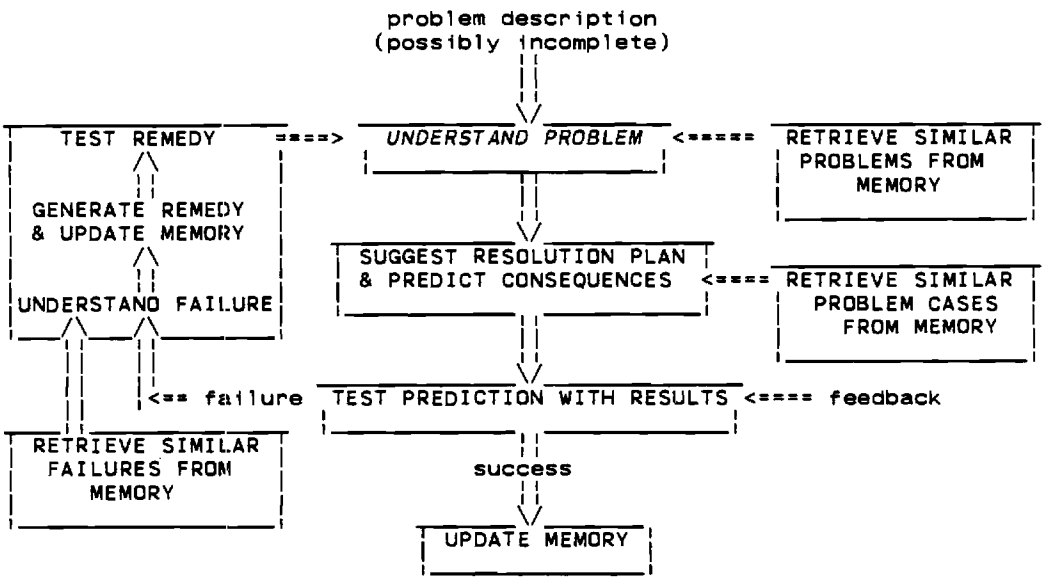


Figure 3-1

Our approach to problem understanding has much in common with AI work in the conceptual information processing of natural language (e.g., Cullingford, 1981; Dyer, 1983; Schank, 1972). Within this framework, an internal representation of the conceptual content of a text is constructed by parsing techniques that primarily key off semantic knowledge. An important part of this approach is the specification of what knowledge an understander uses to fill in missing details and make predictions about the text. We see problem understanding as a more general, but essentially similar, process to natural language processing that is also concerned with the construction of internal representations and their elaboration. When a problem description is confined to text only, then a natural language component is a significant part of problem understanding (Bobrow, 1968; Hayes and Simon, 1979). In general, however, problem descriptions can be in any modality. A problem solver always depends on some interface, be it aural, visual, tactile, or textual for input. However, just as natural language processing requires more than a lexicon and syntax; problem understanding, regardless of the modality, requires much more than surface feature analysis to interpret problems in the environment.

Our approach to problem understanding assumes that a problem solver's internal problem representation is heavily dependent on his domain knowledge and experience. This type of reasoning is analogous to a natural language understanding process that interprets text based on its accumulated domain experience (e.g., Lebowitz, 1980). Variations in either domain knowledge or experience can materially affect the content of a problem representation. This naturally has a direct bearing on the solution, since it is clear that the appropriate representation of problems is crucial to their ultimate solution (Bobrow, 1968; Hayes and Simon, 1979; Polya, 1945).

Understanding a problem, in our view, involves a three stage process: (1) an interpretive process which is responsible for creating an initial coherent problem representation, (2) a conceptually-driven classification process which reformulates the initial representation in terms of known problem types, and (3) an elaboration process that infers important details necessary to problem resolution, but missing from the given problem specification.* Each stage of problem understanding is responsible for some change or addition to the internal problem representation. The net effect of these processes is the construction of a specialized, elaborated problem representation that is available for further reasoning and planning. Our overall process model of problem understanding is reflected in Figure 3-2:

---

*This staged view of problem understanding is analogous to part of the processing performed in Kolodner's (1984) presentation of event reconstruction for fact retrieval.

OVERALL PROBLEM UNDERSTANDING PROCESS

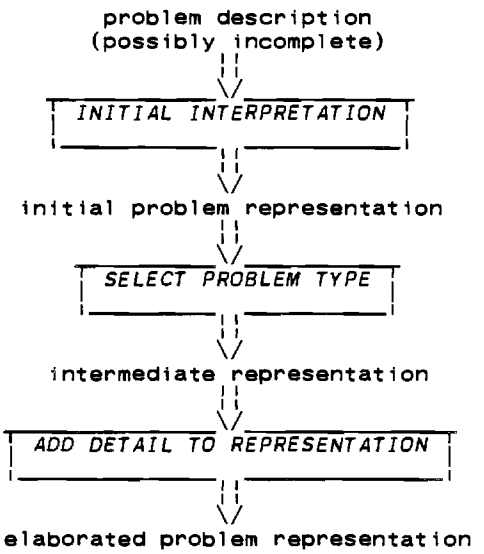


Figure 3-2

While our presentation of problem understanding will describe the control flow in a sequential manner, this is primarily for organizational purposes. Whether the processing is sequential, parallel or interleaved, the functions we describe are essential to the problem understanding task.*

3.2.1 Initial interpretation

Problem understanding begins, with an initial encoding of some situation in terms of preexisting concepts. A problem solver needs an initial interpretation process as an interface to encode the external environment. The initial miscoding of information is the source of many difficulties in understanding.** While we realize the impact initial miscoding can have on the understanding of problems, our emphasis is on the two later stages of understanding illustrated in figure 3-2. Thus, one of our simplifying assumptions is that an initial, possibly incomplete but basicly correct, representation of the problem has already been produced and is now available for further processing.

For the MEDIATOR computer program, we construct the initial representation by hand to simulate this initial interpretation process. We will illustrate the initial interpretation process with the example of Orange-Dispute-0 shown below. In this diagram, the problem description is represented by the sentence, "Two sisters are quarreling over possession of an orange." The hand-constructed initial interpretation is shown in Figure 3-3 below the the box in a structured list format similar to conceptual dependency (Schank and Reisbeck, 1981).

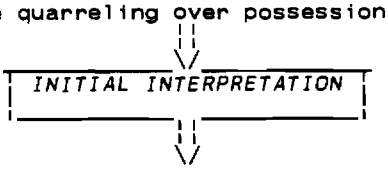
---

*We suspect, because of our analogy to natural language processing, that a flexible control structure is required to allow a kind of "demand driven understanding" of the sort suggested by Schank and Birnbaum (1980) or Granger, Eiselt, and Holbrook (1984). As we will show later, case-based reasoning supports such a "demand driven" approach.

**When people interpret text or events, for example, we know their interpretations are affected by stress, expectations, or other activities during processing. All of these factors lead to wide individual differences in the initial encoding of information. Some of the relevant psychological research includes Bransford, et al. (1972); Loftus (1979); Loftus and Zanni (1975); Neisser (1981); Sulin and Dooling (1974).

SIMULATED INITIAL INTERPRETATION OF "ORANGE-DISPUTE-0"

"Two sisters are quarreling over possession of an orange."




```
(*DISPUTE*
    (PARTY-A SISTER1)
    (PARTY-B SISTER2)
    (DISPUTED-OBJ ORANGE1)
    (ARG-A (*ARGUMENT*
              (ARGUER SISTER1)
              (SUPPORT (*PHYS-CONTROL* (ACTOR SISTER1)
                                       (OBJECT ORANGE1))))
    (ARG-B (*ARGUMENT*
              (ARGUER (SISTER2))
              (SUPPORT (*PHYS-CONTROL* (ACTOR SISTER2)
                                       (OBJECT ORANGE1)))))
```

Figure 3-3

While we do not address initial interpretation issues directly, several points need to be emphasized. First, the initial interpretation process must know the structure and content of the "problem" concept (in our case the "dispute" concept). This knowledge is necessary to ensure that components are related properly in the initial representation. For example, the "dispute" frame provides for a parsimonious "explanation" connecting multiple agents, a physical object, and the "conflict" concept that would be part of the natural language input. Second, phrasal triggers such as "quarrel over" or "fight about" need to explicitly suggest the "dispute" frame as one coherent explanation for the input (e.g., Charniak, 1983; Wilensky, et al., 1984).*

3.2.2 Problem classification

After an initial problem representation is constructed, a problem classification process is responsible for identifying the problem type. A problem type or classification guides the selection of abstract plans and provides a "context" within which to elaborate the representation. Once a problem type is determined, the problem representation is reformulated to reflect this classification. This part of problem understanding is similar to what some consultation models of classification problem solving refer to as "forming a hypothesis" (Weiss and Kulikowski, 1979). The result of this process is an intermediate representation of the initial problem, which includes named plans that a problem solver believes appropriate for problems of that type. This classification decision also constrains later problem elaboration.**

We can illustrate how classification decisions guide planning and demonstrate exactly what we mean by problem classification with the example in Figure 3-4. In this figure, we continue to illustrate the process of understanding Orange-Dispute-0 after the initial interpretation presented above. Using that representation, the dispute is classified as a "physical dispute" (we will explain how later) and the representation is altered as shown below. Those portions of the new representation which provide direction to later planning are in boldface type.

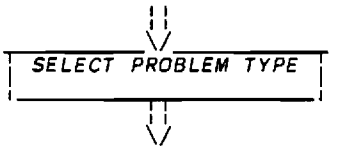
---

*We see the objectives of the problem interpretation phase as being synonymous with those ascribed to text comprehension, i.e., coherence, concreteness, least commitment, and parsimony. Refer to Wilensky (1983) and Greeno (1977) for discussions.

**This is similar to the idea of "constraint propagation" (Stefik, 1981) in planning, where old constraint decisions are used to later refine a planner's options.)

AN EXAMPLE OF DISPUTE CLASSIFICATION

```
(*DISPUTE*
    (PARTY-A SISTER1)
    (PARTY-B SISTER2)
    (DISPUTED-OBJ ORANGE1)
    (ARG-A (*ARGUMENT*
              (ARGUER SISTER1)
              (SUPPORT (*PHYS-CONTROL* (ACTOR SISTER1)
                                       (OBJECT ORANGE1))))
    (ARG-B (*ARGUMENT*
              (ARGUER (SISTER2))
              (SUPPORT (*PHYS-CONTROL* (ACTOR SISTER2)
                                       (OBJECT ORANGE1)))))
```




```
(*PHYS-DISPUTE*
    (USUALLY-USEFUL-PLANS (DIVIDE-EQUALLY TAKE-TURNS
                           DIVIDE-UNEQUALLY))
    (OTHER-PLANS (APPLY-STANDARD-PLAN BINDING-ARBITRATION
                  GAME-OF-CHANCE GAME-OF-SKILL))
    (PARTY-A SISTER1)
    (PARTY-B SISTER2)
    (DISPUTED-OBJ ORANGE1)
    (ARG-A (*ARGUMENT*
              (ARGUER SISTER1)
              (SUPPORT (*PHYS-CONTROL* (ACTOR SISTER1)
                                       (OBJECT ORANGE1))))
    (ARG-B (*ARGUMENT*
              (ARGUER (SISTER2))
              (SUPPORT (*PHYS-CONTROL* (ACTOR SISTER2)
                                       (OBJECT ORANGE1)))))
```

Figure 3-4

As shown above, the original "dispute" has been reinstantiated as a "physical dispute." This represents a further specialization of the original problem.* Figure 3-4 also provides an example of the additional knowledge made available as a result of this classification decision. Because the dispute was classified as a "physical dispute", it inherits, by default, the general plans believed applicable in this context. Notice that the plans are partitioned into two sets. This provides the opportunity to influence the order of plan consideration during later reasoning. The plans identified by the slot "usually-useful-plans" are the first to be evaluated by the MEDIATOR. If none of these general plans are applicable, then those returned via the "other-plans" slot provides a more exhaustive list of alternatives for further consideration. Note also that even within these groupings, the plans can be ordered to control the initial direction of reasoning. For example, the "physical dispute" frame above suggests first evaluate the "divide-equally" plan, then "take-turns", "divide-unequally", etc.

An explicit, declarative, representation of the default planning order provides two types of flexibility. First, by associating the default order of planning with different problem types, we make explicit the relationship between understanding and planning behavior. A problem solver need not consider plans in the same static order for all problem types. Second, by providing a declarative list of plans, we make it possible to dynamically alter the order as a result of experience. If a mediator has had an unusual number of physical dispute cases resolved by "divide-unequally", for example, then that plan could be moved to head the list of "usually-useful-plans" for that class of disputes. In an unorganized rule base, locating all the applicable rules and reordering them to provide this kind of flexibility is quite difficult (e.g., Hayes-Roth et al., 1983; Rychener, 1983). As will be shown later, the direct availability of previous cases provides even more planning flexibility.

---

*The original decision by the initial interpretation process to represent the unspecified problem description as a "dispute" can also be viewed as a classification action. Viewed in this way, the current process is continuing to classify the problem, but at another level of detail.

3.2.3 Problem elaboration

Even after initial interpretation and problem classification, some portions of the problem representation may remain unspecified. or elaborates the representation. During this process, important components of the problem representation needed for later planning are inferred from other parts of the representation or from other knowledge.

There are several reasons why problem representations are not complete. First, this may be a result of the fact that problem descriptions produced for communication to people are intentionally terse. Inference is necessary to fill in the missing details. Another reason that some details of a representation may not be specified is because the information necessary to infer those details has not been derived. A reasoning system which delays its decisions until all the necessary information is available exhibits the "principle of least commitment" (Stefik, 1981; Wilensky, 1983). This principle has been recognized as an important design criteria in promoting efficiency in reasoning. In the MEDIATOR, least commitment has been realized in the default reasoning sequence given to the program. For example, if the disputants' goals are not given, the MEDIATDR will always infer them before attempting to infer the goal relationship because the latter decision depends on the former. In some reasoning systems based on heuristic search (e.g., Sussman, 1975), it is difficult to control this sequence of related decision making. This is especially true when the reasoning system has the option to "guess" either decision. When the "least commitment" reasoner finally has to make a decision, elaboration inference is required to make a heuristic guess.

The results of elaborating an intermediate problem representation are illustrated in the example below. In this diagram, the intermediate representation of Orange-Dispute-O, shown in the upper part of the diagram, is altered to reflect the plausible inference that the sisters want the orange to eat (i.e., their goals are *INGEST* goals). The specific portion affected by the elaboration process is shown in the lower half of the diagram in bold type.

AN EXAMPLE OF DISPUTE ELABORATION

```
(*PHYS-DISPUTE*
    (USUALLY-USEFUL-PLANS (DIVIDE-EQUALLY TAKE-TURNS
                           DIVIDE-UNEQUALLY))
    (OTHER-PLANS (APPLY-STANDARD-PLAN BINDING-ARBITRATION
                  GAME-OF-CHANCE GAME-OF-SKILL))
    (PARTY-A SISTER1)
    (PARTY-B SISTER2)
    (DISPUTED-OBJ ORANGE1)
    (ARG-A (*ARGUMENT*
              (ARGUER SISTER1)
              (SUPPORT (*PHYS-CONTROL* (ACTOR SISTER1)
                                       (OBJECT ORANGE1))))
    (ARG-B (*ARGUMENT*
              (ARGUER (SISTER2))
              (SUPPORT (*PHYS-CONTROL* (ACTOR SISTER2)
                                       (OBJECT ORANGE1)))))
```

ADD DETAIL TO REPRESENTATION

```
(*PHYS-DISPUTE*
    (USUALLY-USEFUL-PLANS (DIVIDE-EQUALLY TAKE-TURNS
                           DIVIDE-UNEQUALLY))
    (OTHER-PLANS (APPLY-STANDARD-PLAN BINDING-ARBITRATION
                  GAME-OF-CHANCE GAME-OF-SKILL))
    (PARTY-A (SISTER1 (HAS-GOAL (*INGEST*
                                 (ACTOR SISTER1)
                                 (OBJECT ORANGE1)))))
    (PARTY-B (SISTER2 (HAS-GOAL (*INGEST*
                                 (ACTOR SISTER2)
                                 (OBJECT ORANGE1)))))
    (DISPUTED-OBJ ORANGE1)
    (ARG-A (*ARGUMENT*
              (ARGUER SISTER1)
              (SUPPORT (*PHYS-CONTROL* (ACTOR SISTER1)
                                       (OBJECT ORANGE1))))
    (ARG-B (*ARGUMENT*
              (ARGUER (SISTER2))
              (SUPPORT (*PHYS-CONTROL* (ACTOR SISTER2)
                                       (OBJECT ORANGE1)))))
```

Figure 3-5

In this case, the goals of the disputing sisters have been inferred by a "goal elaboration inference." There are several different ways that the goals of the disputants can be inferred. For example, the goals can be inferred from the disputants' arguments, the disputed object, or from other sources. Sources of elaboration inference will be discussed more in a later section.

3.2.4 Case-based problem understanding

We have found that case-based reasoning supports problem understanding in two ways:

1. *a previous case can suggest plausible problem classifications*
2. *a previous case can suggest features during problem elaboration.*

A recalled case can suggest a classification for the current problem when the features of a new problem cause the problem solver to be reminded of another previously encountered case. The transfer of the problem classification from a recalled case can be viewed as suggesting a hypothesis for the new problem. A problem solver can then attempt to reinterpret or reformulate the new problem as a member of that category of problems. For example, if a recalled dispute was previously classified as a physical dispute, then that classification might be transferred to the current dispute as long as the transfer is consistent with other facts in the dispute.*

---

*Using a retrieved case for heuristic support of classification judgements is the same notion as that described in the context theory of classification by Medin and Schaffer (1978).

Recalled cases are also used in problem understanding to suggest plausible ways of elaborating the problem representation. With a recurrent problem, a remembered instance of the problem will often share the same type of information. This information can often be transferred directly or with minor modification to a new case. For example, if in the recalled case the disputants had "ingest" goals, then one plausible inference for elaborating the goals of the current disputants would be to transfer and instantiate the same type goals. Even so, the proposed transfer needs to be checked for consistency with previous inferences and general domain knowledge.

These two heuristic aids to problem understanding made possible by case-based reasoning are indicated graphically in the Figure 3-6:

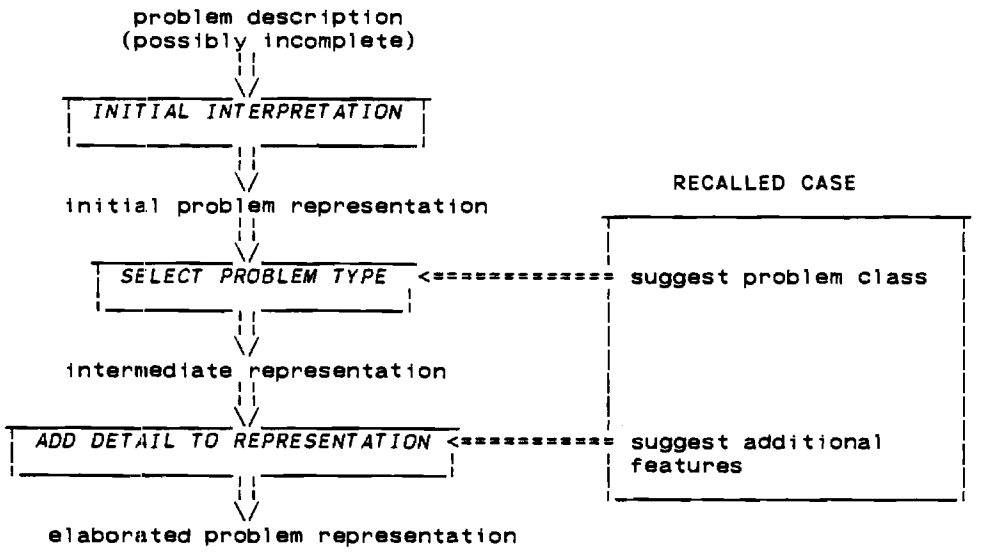


Figure 3-6

3.3 Problem classification

Classification decisions are important to the ultimate solution of problems in two ways. First, the decision helps focus the problem solver's reasoning, since specific plans useful on different problem types are made available. Second, a classification "hypothesis" influences or colors later elaboration decisions, allowing the problem representation to evolve as a coherent unit (Greeno, 1977; Wilensky, 1983).

Depending on the domain, a suggested classification hypothesis may or may not be subjected to extensive consistency checking. For example, in the mediation domain the "cost" of misunderstanding a dispute is, in general, not that great. Usually this means nothing greater than a potential delay in resolving the dispute. The worst that normally occurs is that the dispute does not get resolved. For most disputes, this certainly is bad, but not tragic. Consider the difference, however, if the problem solver is a medical consultant performing a disease diagnosis. In this situation, the hypothesis should be subjected to consistency and exclusion checks, since a life could depend on the decision.

3.3.1 A case-based classification algorithm

Making classification decisions based on previous cases is not intended to replace other means of making such decisions. Instead, it is a means of augmenting these invariant lines of reasoning. The following algorithm illustrates how we incorporate analogy to previous cases into the classification process of problem understanding.

A CASE-BASED PROBLEM CLASSIFICATION ALGORITHM

1. Recall similar cases and select the one most similar to the current case.

2. If there is a similar case, then check the applicability of the classification (depends on the judged degree of risk) obtained from the recalled case. If the classification is applicable, then transfer the classification and reformulate the problem as an instance of the transferred type.

3. Otherwise, classify the problem by default reasoning.

Figure 3-7

Let us look at how this algorithm is realized in the dispute mediation domain. The first step, the recall and selection of the most similar previous case, has been outlined in chapter one and will be discussed at length in chapter six. In the mediation domain, this retrieval and selection process is designed to encourage the recall of cases which have the same goals and goal relationship as the current case. This is because it is the goals of disputants that determine appropriate planning strategies. This presents a problem, however, since the mediator (problem solver) does not necessarily know the goals of the disputants at that point. The goals are inferred as part of a later elaboration process. This leads to the following circularity in reasoning: the goals are needed to help choose the best case, so that the best case can be used to infer the problem class, which can be used to infer the disputant's goals.*

Our solution to avoiding such deadlocks is to explicitly order these decisions. Classification decisions precede elaboration inferences. In the absence of known goal information, classification is based on whatever is available in the initial representation. In some disputes, the best available information is the identification of the disputants, their arguments and the disputed object. Step two of the classification algorithm above directs the transfer of the classification from the most similar case when judged applicable. In the MEDIATOR, the transfer is automatic unless explicitly inhibited. This is an implicit estimate that the degree of risk involved in a misclassification is minimal. Thus, the classification depends on the most similar recalled case and the most similar case, in turn, is determined by an evaluation of information available in the initial representation.

The following fragment from the MEDIATOR program illustrates this classification algorithm as it processes Orange-Dispute-0, introduced earlier. In this instance, the case occurs in sequence after the MEDIATOR has resolved Candy-Dispute-0, so we expect it to be reminded of that case. With that reminding, the orange dispute is similarly classified, even though the goals of the disputants are not yet explicitly known.

---

*This type of circularity is one of the insidious problems that is difficult to detect in rule-based systems. One instance of this is known as the "least commitment deadlock" where one set of rules is waiting on another set before either will commit (Hayes-Roth et al., 1983). See Stefik (1981) for another solution to this problem.

I/O BEHAVIOR OF THE MEDIATOR DURING CASE-BASED DISPUTE CLASSIFICATION

```
(mediator orange-dispute-0 t)
Considering the following problem:
   two sisters are quarreling over an orange,
   which has been presented as ako M-DISPUTE.
(*DISPUTE*
 (PARTY-A (SISTER1))
 (PARTY-B (SISTER2))
 (DISPUTED-OBJ ORANGE1)
 (ARGUMENT-A (*ARGUMENT* (ARGUER (SISTER1))
                         (SUPPORT (*PHYS-CONTROL* (ACTOR SISTER1)
                                                  (OBJECT ORANGE1)))))
 (ARGUMENT-B
  (*ARGUMENT* (ARGUER (SISTER2))
              (SUPPORT (*PHYS-CONTROL* (ACTOR SISTER2)
                                       (OBJECT ORANGE1))))))
ATTEMPTING TO RECALL SIMILAR PROBLEMS IN ORDER TO CLASSIFY THIS ONE...
  looking for disputes with similar disputants or with similar goals...
  looking for disputes whose disputants used similar arguments.
  looking for disputes involving similar objects...
   reminded of two boys are quarreling over a candy bar
    because the object in that case, CANDY1, and ORANGE1
    are both of type M-FOOD.
   reminded of two boys are quarreling over a candy bar
      because the argument used in that case, the possession argument,
                                 was of the same type, M-POSSESS, as this dispute.
There was one previous case found.
   #<M-PHYS-DISPUTE 2213i722> was the case where two boys are quarreling
        over a candy bar.
Selected the dispute where two boys are quarreling over a candy bar
   which was classified as M-PHYS-DISPUTE for further consideration.
Transferring previous classification to this dispute since the
   disputed object is a M-PHYS-OBJ.
This dispute will be referred to as #<M-PHYS-DISPUTE 22475211>
(*PHYS-DISPUTE*
 (PARTY-A (SISTER1))
 (PARTY-B (SISTER2))
 (DISPUTED-OBJ ORANGE1)
 (ARGUMENT-A (*ARGUMENT* (ARGUER (SISTER1))
                         (SUPPORT (*PHYS-CONTROL* (ACTOR SISTER1)
                                                  (OBJECT ORANGE1)))))
 (ARGUMENT-B
  (*ARGUMENT* (ARGUER (SISTER2))
              (SUPPORT (*PHYS-CONTROL* (ACTOR SISTER2)
                                       (OBJECT ORANGE1))))))
```

Figure 3-8

3.3.2 Default classification

If no previous case is recalled or if the previous case is judged not applicable by domain dependant criteria, then default reasoning is required. This is the third step in the classification algorithm above. In general, there are many ways problem classification can be performed. Often, a set of rules is used to form a "classification decision tree." Classification reasoning, in these systems, is a process of searching this rule tree to find one or more classifications consistent with all the rules in its path (e.g., Pople, 1977; Weiss and Kulikowski, 1979). This is the type of static reasoning that case-based reasoning is intended to by-pass. We cannot, however, replace this default reasoning because there may be times when no previous case is available or applicable.

In the MEDIATOR, default classification of disputes is performed by the simple heuristic below:

DEFAULT CLASSIFICATION OF DISPUTES

1. If there is no suggested classification, then choose the physical dispute classification as long as the disputed object is a physical object.

2. If there are several possible classifications suggested, then order the suggestions by the following ranking physical disputes >> economic disputes >> political disputes.

Figure 3-9

When we defined the three types of disputes, we realized that some disputes could be viewed as members of several dispute classifications. We have attempted to minimize this inconsistency by choosing to bias default classification toward physical disputes (see section 2.3.6). If there is no previous case recalled from memory, and the object of dispute is a physical object, the physical dispute context is the default classification. This heuristic is based on the observation that disputes usually involve, no matter how incidentally, some physical object or set of objects. In the absence of evidence, we might as well begin with the most basic (i.e., physical) interpretation before looking at more complex classifications. Some evidence that might cause a mediator to choose an economic classification instead include the fact that the disputed object is the "price" or "value" of an object and one of the disputants has a "merchant" role theme.

Default classification is also required in selecting the best case when more than one is recalled from memory. This is where the second part of the above heuristic applies. For example, when there is more than one classification suggested among the recalled cases and there is no evidence to suggest any preference (i.e., they are all equally rated by a heuristic evaluation function to be described later), then the "best" case is selected based on the following default order: (1) physical, (2) economic, and (3) political.

This second classification decision is behind the behavior illustrated in the following sample computer output. In this situation, the MEDIATOR is asked to resolve Candy-Dispute-0. The only cases in memory at this time are the Panama Canal dispute and the Korean conflict. Obviously, these cases are only superficially similar to the candy dispute. The point of this exercise was to begin training the MEDIATOR by building up its case experience. Default classification is especially important in this situation. It can be thought of as analogous to the first time a novice puts his book training to the test on a real problem. With no previous experience to help guide him, the novice has to rely on only what he has been told (i.e., default reasoning). The point at which default classification occurs is in boldface type.

I/O BEHAVIOR OF THE MEDIATOR DURING DEFAULT DISPUTE CLASSIFICATION

```
(mediator candy-dispute-0 t)
Considering the following dispute problem:
   two children are quarreling over a candy bar,
   which has been presented as ako M-DISPUTE.
ATTEMPTING TO RECALL SIMILAR DISPUTES IN ORDER TO CLASSIFY THIS ONE
  looking for disputes with similar disputants or with similar goals
  looking for disputes whose disputants used similar arguments...
  looking for disputes involving similar objects...
   reminded of the case where the US and Panama are quarreling over
      the Canal because both disputants were of type M-PARTY.
   reminded of the case where Korea was in dispute because the
      object in that case, Korea, was of a type similar to candy1.
There were two previous cases found.
   #<M-POL-DISPUTE 21016670> was the case where the US and Panama
                             are quarreling over the Canal.
   #<M-PHYS-DISPUTE 21016524> was the case where Korea was in dispute.

Using default evaluation function, ranking cases based on the three
   invariance features disputant arguments, objects, and disputants.
#<M-PHYS-DISPUTE 21016524> is chosen as the most analogous case to
   #<M-DISPUTE 21016135> based on these criteria.
Selected the dispute where Korea was in dispute
   which was classified as M-PHYS-DISPUTE for further consideration.
Transferring previous classification to this dispute.
This dispute will be referred to as #<M-PHYS-DISPUTE 21034361>
```

Figure 3-10

In this case, the MEDIATOR recalled a political dispute (the Panama Canal dispute) and a physical dispute (the Korean conflict). Both these disputes were rated for similarity to Candy-Dispute-0 by a default evaluation function (the evaluation function will be explained in chapter six) according to three features: "disputant arguments," "disputed objects," and "disputant types." Although the output does not show the ratings, both remindings were rated equally low, as might be expected. In this situation, the physical dispute classification gets the choice in accordance with the preceding heuristic.

3.4 Case-based and other elaboration inferences

Some problem components may not be filled in as a result of initial interpretation and classification. In this situation, the problem solver has to infer plausible fillers for portions of the problem representation important to later reasoning. This is the general class of inference we call elaboration. To a certain degree, the majority of the work in problem understanding is accomplished by elaboration inferences. There are two issues with respect to elaboration: (1) what components should be elaborated and (2) what sources of information support these inferences?

The first of these two issues, deciding which components to elaborate, is important because a problem solver may have time or other resource constraints which limit this process. For example, parts of the representation may intentionally remain empty, if the "cost" to infer these components is disproportionate to their value to a problem solver. It is for this general reason that critical portions of the representation be given priority for elaboration. This piece of knowledge may be one of the significant differences that separate experts from novices (e.g., Chase and Simon, 1973). The second issue above requires a problem solver to know different sources from which components may be inferred. This is also a subtle point of expertise. It serves to insure that an expert problem solver can succeed where the less competent might fail. When a primary source of inference is unavailable, for example, a good problem solver can still infer an important part of the representation from alternate sources.

It is difficult to specify in general terms what components of a problem representation are the most important or what sources of information provide the best evidence for elaboration. However, we will address these points through illustrations from our dispute mediation domain. First, we will discuss those components that need elaboration.

An analysis of a problem solver's reasoning will usually reveal a chain of inferences that begins with whatever information is presented explicitly in the initial problem representation. The initial focus of this chain of inferences for a given problem should be the most important component of the problem representation not specified directly. For example, in the dispute mediation domain, the disputants' real goals are rarely specified, but are the single most important component of the dispute representation since they allow important inferences to be made (e.g., the goal relationship and the mediation plan). In what follows, we will focus on how the disputants' goals can be inferred by different types of elaboration. We have identified three types of elaboration which support the inference of disputants' goals. These are: (1) default, (2) thematic, and (3) direct goal elaborations.

Default elaboration is an indirect means of inferring a disputant's goals or other portions of representations. It is accomplished by organizing related goal information so that it explicitly fills specific components in related knowledge structures (e.g., interpersonal themes). When these knowledge structures are instantiated, the goal information is automatically provided by default. Default elaboration supports problem understanding and problem solving in two ways: (1) it insures that the representation is consistent for later reasoning and (2) it influences a problem solver according to the underlying domain model. As you will recall in the baby dispute, the women both claim to be the baby's mother (a thematic argument). The "mother" interpersonal theme represented within both women's argument representation is elaborated with the default knowledge that this theme normally evokes protecting goals and caring goals. In this way, the representation of a theme always includes the default goal knowledge which is consistently available for later direct elaboration as needed. Default elaboration also influences the problem solver according to the underlying "beliefs" of the domain model. For example, again using the default goal information provided by the "mother" theme, direct goal inferences can decide how best to elaborate the representation of a disputant's goals. Default elaboration, in this situation, "predisposes" the disputant's representation, hence the planner, in the direction of believing that a mother wants to protect and care for her baby.

Another indirect means of inferring disputants' goals is from their known interpersonal relationships or roles they occupy (Schank and Abelson, 1977). We call this "thematic" elaboration. This type of elaboration is responsible for filling in specific thematic (e.g., "role-theme" slot in M-PERSON frame) in the MEDIATOR's representation of a disputant. We provide a strategy for thematic elaboration in the following subsection. One source of thematic knowledge is the disputant's argument. When womani argues, in the baby dispute, that she is the baby's mother, we can infer that the interpersonal theme "mother" should be part of the representation of womani. Because themes often provide information that describes the relationship between or among several entities, it is sometimes necessary to alter the representations of related components in order to enforce certain constraints in the domain model. For example, once we infer that womani is the mother of babyi, we must also modify the representation of babyi to reflect the complementary "child" interpersonal theme. This provides the expected interpersonal information identifying womani as the baby's mother and enforces a constraint that the concepts "baby" and "mother" must be co-referential.

The final way that a disputant's goals can be inferred is by direct goal inference. By this we mean taking a goal representation from some other part of the problem and inferring that it is a disputant's goal. This elaboration inference often depends on other inferences and constraint knowledge. To illustrate its dependence on earlier inference, recall that one way to infer a disputant's goal is from known thematic information. Thus, if woman1 has the "mother" interpersonal theme with baby1, then we can make the direct inference that woman1 has the protection and nurture goals expected by default of a mother. This goal representation can then be moved from the thematic component to the goal component.

A problem solver must be careful in doing elaboration to maintain consistency while individual pieces of the representation are being inferred. For example, according to the above elaboration inferences both women in the baby dispute will be inferred to be the baby's mother. We need some way to detect this contradiction and infer alternate goals for one of the women. We will present such a method of detecting elaboration errors in subsection 3.4.3.

Notice that goal elaboration can be dependant on thematic elaboration which can depend on default elaboration. This illustrates a natural dependency between some of the different types of elaboration. Knowledge of this dependency, while useful in the elaboration stage of problem understanding, becomes especially important in failure recovery since locating the source of misunderstanding generally involves some sort of dependency-directed reasoning (e.g., Doyle, 1979; O'Rorke, 1983; Sussman and Stallman 1979). The use of this dependency information is discussed in chapter five.

This brings us to the second elaboration issue, the sources of information necessary to support these inferences. Sources of information vary by domain, but they can usually be arranged in a hierarchy of preferred sources.* For example, in dispute mediation we use the following preference hierarchy of sources to infer disputants' goals:

1. the disputants' arguments
2. a recalled similar case
3. the disputed object.

This preference order reflects our observation that disputants often attempt to justify their action or position by asserting specific information in their argument that can be used to infer their goals. For example, recall the arguments used by the women in the baby dispute. It is for this reason that we consider the disputants' arguments as the preferred source of goal inference. A recalled similar case is only second best since it is outside of the current case. In the absence of an argument, however, a similar case can provide specific goal inferences, depending on the degree of similarity between the cases. For example, in Orange-Dispute-0, the reasoner infers the goal from the related component from Candy-Dispute-0. The last source of evidence is the disputed object. It is considered the least reliable because of the many ways that some objects can be used. Disputed objects, while often relatively unreliable, do provide an alternative source of goal inference in some cases. For example, in Candy-Dispute-0 the mother-mediator inferred the boys' goals from the disputed object, the candy bar. We will discuss each of the above sources of elaboration inference in the following sections.

---

*Here we are appealing to the same notion as "best evidence" in legal reasoning. A trivial example is that an eyewitness provides a better source of evidence than a hearsay witness.

3.4.1 Elaboration based on disputant arguments

From a worst case perspective, we cannot expect more in the initial representation than some idea of the disputants and the disputed object. For example, "Two women are quarreling over a baby," is a minimal problem description of the baby dispute. If the disputants provide arguments on their behalf or in opposition to their competitor, we should capitalize on this additional source of inference.

Disputant arguments can potentially provide the most direct source of evidence in inferring disputant goals. This can happen in several ways. The disputant can include his goal as justification in an argument or as support for his position in the dispute. A disputant's goals may also be inferred from his opponent's argument. This can happen when the opponent attacks the disputant and offers an alternative motivation to explain the weakness of the disputant position. In addition, interpersonal and role themes can also be inferred analogously from the arguments of the disputants. We have developed the elaboration strategy shown in Figure 3-11 to focus on goal and theme inference from disputants' arguments:

AN ELABORATION STRATEGY BASED ON DISPUTANT ARGUMENTS

1. *Goals* for a disputant can be inferred from the disputant's argument as follows:
   a. If the supported point in the argument is a goal and the actor of the goal is the disputant, infer that goal is the disputant's goal.
   b. If the support for the argument is a goal and the actor of that goal is the disputant, then infer that goal to be the disputant's goal.
2. *Goals* for a disputant can be inferred from his opponent's argument as follows:
   a. If other inferences have failed or have resulted in contradictions and the other disputant argument includes a goal assertion as the opponent's point and the actor of the goal is the disputant, then infer that goal as the disputant's goal.
   b. If other inferences have failed or have resulted in contradictions and the other disputant argument includes a goal assertion as the attack portion of the argument and the actor of the goal is the disputant, then infer that goal as the disputant's goal.
3. *Themes* for a disputant can be inferred from a disputant's argument by an analogous process to that for goals in 1 and 2 above.
4. If the argument is recognized as a persuasive force argument, then make no inferences based on the argument.

Figure 3-11

The following examples illustrate how this strategy is employed. First, assume that one of the boys in the candy dispute makes the following argument on his behalf: "I should get the candy because I want to eat it." We represent this argument as:

```
(*ARGUMENT* (ARGUER BOY1)
            (SUP-POINT (M-INGEST (ACTOR BOY1)
                                 (OBJECT CANDY1)))
            (SUPPORT (*PHYS-CONTROL* (ACTOR BOY1)
                                     (OBJECT CANDY1)))
            (OPP-POINT NIL)
            (ATTACK NIL))
```

Using this representation, step i.a of the strategy above allows us to infer that boy1 has an ingest goal. Note the difference, however, if the argument had been the following: "I want to give the candy to Mary, so she can eat it." This argument is represented in the following manner:

```
(*ARGUMENT* (ARGUER BOY1)
            (SUP-POINT (M-INGEST (ACTOR MARY)
                                 (OBJECT CANDY1)))
            (SUPPORT (M-ATRANS (ACTOR BOY1)
                               (OBJECT CANDY1)
                               (TO MARY)
                               (FROM BOY1)))
            (OPP-POINT NIL)
            (ATTACK NIL))
```

We avoid inferring that boy1's goal is to ingest the candy by noting that the actor in the "sup-point" slot, Mary in this case, is not the disputant. Using step 1.b of the strategy, however, lets us infer the boy's intention to give the candy to Mary by next examining the "support" slot of the argument.

Part two of the elaboration strategy shown in Figure 3-11 says that you can sometimes infer someone's goal by listening to their opponent. Obviously this heuristic has limited value, but it did come into play in the baby dispute. Solomon knew that both women could not be the baby's mother, so what could be motivating the other woman? According to the story, both women accused the other of wanting to replace the dead baby with the living child in order to preserve her maternal status. This means that once Solomon decided that it was inconsistent for both women's arguments to hold, he had no other information from which to infer the goal of the other woman besides her opponent's argument. Thus, step two above allows goals to be inferred from the opponent's argument when there are no other options available. In the baby dispute, woman1's argument is represented as shown below:

```
(*ARGUMENT* (ARGUER (WOMAN1))
            (SUP-POINT (*PHYS-CONTROL* (ACTOR WOMAN1)
                                       (OBJECT BABY1)))
            (SUPPORT (*IPT-MOTHER* (ACTOR WOMAN1) (CHILD BABY1))
                                   (EXPECT-GOAL (PROTECT CARE-FOR))))
            (OPP-POINT (*PHYS-CONTROL* (ACTOR WOMAN2)
                                       (OBJECT BABY1)))
            (ATTACK
               (LEADTO (ANTE (BABY2 (HEALTH -10)))
                       (CONSEQ (*SUBSTITUTE* (ACTOR WOMAN2)
                                             (OBJECT BABY1)
                                             (FOR-OBJ BABY2)
                                             (INST-TO
                                                PRESERVE-STATUS-QUO))))))
```

This representation stands for woman1's thematic argument in support of her gaining physical control of the baby, and attacking woman2's point by asserting that woman2 wants to substitute baby1 for the dead baby. In this case, the final goal inference came via the default inference that "substitution" actions are normally instrumental to preserving the preconditions or status quo of an actor (i.e., a precondition to being a mother is having a child; substituting another child for a dead child preserves that precondition). On yet another level, the social stigma attached to a mother who fails to properly care for her child is significant. Substitution of another child avoids this social punishment and preserves her social status quo.*

Part three of our argument-based elaboration strategy permits indirect goal inference by first inferring applicable thematic relationships from the disputants' arguments. Thus, in the baby dispute, the women offered maternal thematic arguments on their behalf. Once a theme has been inferred, its goal expectations are available as plausible goals for a disputant, as was illustrated above.**

---

*This is similar to Wilensky's (1983) social relationship subsumption state. Thus substitution could be viewed as a plan to restore a negated social relationship (i.e., motherhood) with its associated recurring child care goals.

**This is a specific application of the idea behind the "invoke theme" planbox that was part of the "persuade" package in Schank and Abelson (1977). We do not address the problem of multiple goal resolution when a disputant has more than one theme. See Wilensky (1983) for a discussion of this.

3.4.2 Elaboration using objects and previous cases

Once a mediator has elaborated the problem representation from argument-based inferences, the other primary sources of elaboration are previous cases and the disputed object. Because the minimal description of a dispute need only specify the disputants and the disputed object, previous cases or prior knowledge of the normal uses of the disputed object may be the only source of information that allows problem elaboration. For example, given the following simple description of Orange-Dispute-0: "Two sisters are quarreling over an orange," how do we infer the sisters' goals? We have developed the elaboration strategy shown in Figure 3-12 to permit such inference.

AN OBJECT AND CASE-BASED ELABORATION STRATEGY

1. Using the most similar case recalled from memory
2. If the disputant's goal is unknown and there are no similar cases recalled from memory, then
   a. Infer the goal from the default use of the disputed object for this classification.
   b. Otherwise, recall the object from experience that is most similar to the disputed object and infer its normal use in this type of problem as the disputant's goal.
3. Otherwise, transfer the goal type from the corresponding disputant in the recalled case and instantiate the same goal for the current disputant.
4. Finally, check all inferred goals for consistency with the default use of the disputed object within this classification.

Figure 3-12

Even though our preferred mode of reasoning is case-based, part two of this strategy allows the mediator to infer goals via default reasoning in the absence of useful experience. Problem classification, as established by the previous stage of understanding, is an important part of default elaboration reasoning since it is context-dependent. For example, if we infer that the dispute is a physical dispute, then the default use for an orange is "ingestion" and the disputant an "ingest" goal. If the context were changed to an economic one, however, the default use for an orange becomes "commercial" (i.e., ATRANS for money). Step 2.b of this strategy allows the problem solver to use previous experience with any similar object as a source of inference for the disputant's goal.

Step three of this strategy represents the case-based approach to goal inference. Instead of inferring the disputant's goal from the disputed object, the corresponding goal of the disputant from the recalled case is considered for transfer. For example, if the mediator recalls Candy-Dispute-0 when elaborating the goals of the sisters in Orange-Dispute-0, then the sisters' goals can be transferred from Candy-Dispute-0. This is illustrated in the following program fragment which continues Orange-Dispute-0 presented earlier. The first portion of this example repeats the earlier classification shown in Figure 3-13. The portion where case-based elaboration is occurring is indicated in bold type.

I/O BEHAVIOR OF THE MEDIATOR DURING CASE-BASED ELABORATION

```
(mediator orange-dispute-0 t)
Considering the following problem:
   two sisters are quarreling over an orange,
   which has been presented as ako M-DISPUTE.
(*DISPUTE*
 (PARTY-A (SISTER1))
 (PARTY-B (SISTER2))
 (DISPUTED-OBJ ORANGE1)
 (ARGUMENT-A (*ARGUMENT* (ARGUER (SISTER1))
                         (SUPPORT (*PHYS-CONTROL* (ACTOR SISTER1)
                                                  (OBJECT ORANGE1)))))
 (ARGUMENT-B
  (*ARGUMENT* (ARGUER (SISTER2))
              (SUPPORT (*PHYS-CONTROL* (ACTOR SISTER2)
                                       (OBJECT ORANGE1))))))
```

```
ATTEMPTING TO RECALL SIMILAR PROBLEMS IN ORDER TO CLASSIFY THIS ONE...
  looking for disputes with similar disputants or with similar goals...
  looking for disputes whose disputants used similar arguments.
  looking for disputes involving similar objects...
   reminded of two boys are quarreling over a candy bar
    because the object in that case, CANDY1, and ORANGE1
        are both of type M-FOOD.
   reminded of two boys are quarreling over a candy bar
      because the argument used in that case, the possession argument,
                          was of the same type, M-POSSESS, as this dispute.
There was one previous case found.
   #<M-PHYS-DISPUTE 22131722> was the case where two boys are quarreling
        over a candy bar.
Selected the dispute where two boys are quarreling over a candy bar
   which was classified as M-PHYS-DISPUTE for further consideration.
Transferring previous classification to this dispute.

This dispute will be referred to as #<M-PHYS-DISPUTE 22475211>
(*PHYS-DISPUTE*
 (PARTY-A (SISTER1))
 (PARTY-B (SISTER2))
 (DISPUTED-OBJ ORANGE1)
 (ARGUMENT-A (*ARGUMENT* (ARGUER (SISTER1))
                         (SUPPORT (*PHYS-CONTROL* (ACTOR SISTER1)
                                                  (OBJECT ORANGE1)))))
 (ARGUMENT-B
  (*ARGUMENT* (ARGUER (SISTER2))
              (SUPPORT (*PHYS-CONTROL* (ACTOR SISTER2)
                                       (OBJECT ORANGE1))))))
SISTER1 has presented an argument recognized as type *PHYS-CONTROL*
  which is normally presented in an attempt to persuade by force.
Therefore no inferences will be based on SISTER1's argument.

SISTER2 has presented an argument recognized as type *PHYS-CONTROL*
  which is normally presented in an attempt to persuade by force.
Therefore no inferences will be based on SISTER2's argument.

Attempting to transfer goal type from case #<M-PHYS-DISPUTE 22131722>
   checking for consistency with normal uses of ORANGE1.
Thus SISTER1 is inferred to have a M-INGEST goal
   which is consistent with the normal uses of #<M-ORANGE 22123746>
    in this context.

Attempting to transfer goal type from case #<M-PHYS-DISPUTE 22131722>
   checking for consistency with normal uses of ORANGE1.
Thus SISTER2 is inferred to have a M-INGEST goal
   which is consistent with the normal uses of #<M-ORANGE 22123746>
    in this context.
(*PHYS-DISPUTE*
 (PARTY-A (SISTER1 (*GOAL* (*INGEST* (ACTOR SISTER1)
                                     (OBJECT ORANGE1)))))
 (PARTY-B (SISTER2 (*GOAL* (*INGEST* (ACTOR SISTER2)
                                     (OBJECT ORANGE1)))))
 (DISPUTED-OBJ ORANGE1)
 (ARGUMENT-A
  (*ARGUMENT* (ARGUER (SISTER1 (*GOAL* (*INGEST* (ACTOR SISTER1)
                                                 (OBJECT ORANGE1)))))
              (SUPPORT (*PHYS-CONTROL* (ACTOR SISTER1)
                                       (OBJECT ORANGE1)))))
 (ARGUMENT-B
  (*ARGUMENT*
   (ARGUER (SISTER2 (*GOAL* (*INGEST* (ACTOR SISTER2)
                                      (OBJECT ORANGE1)))))
   (SUPPORT (*PHYS-CONTROL* (ACTOR SISTER2) (OBJECT ORANGE1))))))
```

Figure 3-13

The fourth step of the case-based elaboration strategy presented in Figure 3-12 calls for a consistency check on goal transferal from other cases. This will be discussed in the next section as part of a more general discussion of recognizing elaboration errors.

3.4.3 Recognizing elaboration errors

In elaborating a problem representation, there is a risk that a plausible inference may in fact be in error. We have identified three types of increasingly more complex elaboration errors: (1) consistency errors, (2) contradictions, and (3) undetected errors. Each of these errors can result from any of the elaboration methods discussed above. The detection and avoidance of these errors is especially important to the effective transfer of information from previous cases.

Consistency errors are single inferences that violate a specific portion of the underlying domain model. An example of the checking that happens in an attempt to detect these errors was illustrated in the previous section. An M-INGEST goal was suggested for transfer from Candy-Dispute-0 to Orange-Dispute-0. It was instantiated only after it was verified to be consistent with the known normal uses of oranges in the physical dispute context. This method of blocking a goal transfer is equivalent to the frame-based method of specifying restrictions on the values that can fill a given slot (Minsky, 1975).

To illustrate the use of consistency checks to detect and prevent improper goal inference, we will consider Candy-Dispute-0, previously used to illustrate default classification. The MEDIATOR, in this situation, tries to transfer the goals from its "best" previous case. Consistency checks using the default normal usage of the disputed object within the classification prevent an improper goal transfer. As before, we repeat the previous output portion and highlight the constraint- checking behavior in bold type.

I/O BEHAVIOR ILLUSTRATING CONSISTENCY CHECKING DURING ELABORATION

```
(mediator candy-dispute-0 t)
Considering the following dispute problem:
   two children are quarreling over a candy bar,
   which has been presented as ako M-DISPUTE.
ATTEMPTING TO RECALL SIMILAR DISPUTES IN ORDER TO CLASSIFY THIS ONE
  looking for disputes with similar disputants or with similar goals
  looking for disputes whose disputants used similar arguments...
  looking for disputes involving similar objects...
   reminded of the case where the US and Panama are quarreling over
      the Canal because both disputants were of type M-PARTY.
   reminded of the case where Korea was in dispute because the
      object in that case, Korea, was of a type similar to candy1.
There were two previous cases found.
   #<M-POL-DISPUTE 21016670> was the case where the US and Panama
                             are quarreling over the Canal.
   #<M-PHYS-DISPUTE 21016524> was the case where Korea was in dispute.

Using default evaluation function, ranking cases based on the three
   invariance features dispute plans, objects, and disputants.
#<M-PHYS-DISPUTE 21016524> is chosen as the most analogous case to
   #<M-DISPUTE 21016135> based on these criteria.
Selected the dispute where Korea was in dispute
   which was classified as M-PHYS-DISPUTE for further consideration.
Transferring previous classification to this dispute.
This dispute will be referred to as #<M-PHYS-DISPUTE 21034361>
Attempting to transfer goal type from recalled case
   #<M-PHYS-DISPUTE 21016524> checking for consistency
   with normal uses of candy1.
Transfer judged not appropriate for this case
   because of a mismatch with the normal uses of candy1.
Using elaboration to infer CHILD1's goal
   from normal uses of the disputed object in this context.
Therefore CHILD1 is inferred to have a M-INGEST goal.
Using elaboration to infer CHILD2's goal
   from normal uses of the disputed object in this context.
   normal use of -#<M-CANDY 21015553>- is being assumed.
Therefore CHILD2 is inferred to have a M-INGEST goal.
```

Figure 3-14

The second type of elaboration error is *contradiction*. Contradictions occur when multiple elaboration inferences, which do not in themselves directly violate domain constraints, are mutually exclusive. For example, constraints, are mutually exclusive. For example, mother is, by itself, consistent with Solomon's world knowledge. Individually, each woman could very well be the baby's mother as far as Solomon knows. Taken together, however, these two assertions are contradictory. In the presence of the first assertion, default elaboration results in changes to both the woman and baby's representation. This type of reasoning is analogous to the usual notions of non-monotonic reasoning (e.g., Doyle, 1979). It is equivalent to the statement: *Assume X unless and until X can be disproved.* In order to

test for contradictions, the problem solver needs to know what information would "disprove" the default assumption. In this example, finding that woman1 is believed to be the mother of baby1, while attempting to assert that woman2 is baby1's mother, is sufficient to recognize the contradiction and prevent the elaboration.*

In the MEDIATOR, contradiction detection is accomplished by the execution of special procedures attached to certain domain data types that are responsible for insuring that the evolving representation is not contradictory. These procedures effectively represent the problem solver's consistency knowledge concerning the problem domain. For example, the interpersonal theme slot in the representation of woman2 is elaborated by instantiating the "IPT-MOTHER" theme. A procedural attachment to "IPT-MOTHER" called "if-not-disprovable" is executed at the same time. This procedure is responsible for maintaining the consistency between the representations of mothers and their children. Thus, it inspects the child slot of the mother theme to verify that the corresponding mother slot in the child's representation reference each other. If, as happens in the baby dispute, a contradiction is discovered, then the "IPT-MOTHER" instance is deleted from the representation. This is responsible for the behavior illustrated in the following program fragment:

I/O BEHAVIOR ILLUSTRATING CONTRADICTION RECOGNITION DURING ELABORATION

```
(solomon baby-dispute)
Considering the following dispute problem:
   two women are quarreling over a baby,
   which has been presented as ako M-DISPUTE.
Attempting to recall similar disputes in order to classify this one...
  looking for disputes with similar disputants or with similar goals...
  looking for disputes whose disputants used similar arguments...
  looking for disputes involving similar objects...
There were zero cases found.

Given that there are no similar disputes,
  will use a default physical dispute classification.

This dispute will be referred to as #<M-PHYS-DISPUTE 12237676>

WOMAN1 has invoked a thematic argument to support WOMAN1's point:
   wants to take baby1
   Elaborating representation of WOMAN1 by inferring
   a IPT-MOTHER interpersonal theme.
Checking to see that the mother relationship is consistent
   with what is known about BABY1.
   Inferring that BABY1's mother is WOMAN1

WOMAN2 has invoked a thematic argument to support WOMAN2's point:
   wants to take baby1
   Elaborating representation of WOMAN2 by inferring
   a IPT-MOTHER interpersonal theme.
Checking to see that the mother relationship is consistent
   with what is known about BABY1.
Incompatible inference!
WOMAN1 is currently believed to be BABY1's mother.

   Withdrawing the contradictory interpersonal theme inference.

Using WOMAN1's attack argument to infer that WOMAN2
   wants baby1 to replace her dead baby.
   Elaborating representation of WOMAN2 by inferring
   a M-PRESERVE-PRESTIGE goal.

Using interpersonal theme IPT-MOTHER to infer WOMAN1's goal
Therefore WOMAN1 is inferred to have a M-PRESERVE-HEALTH goal
 (wants to preserve baby1's health).

WOMAN2 is represented as having the goal #<M-PRESERVE-PRESTIGE 12236623>
                          (wants baby1 to replace her dead baby).

Goal relationship is inferred to be COMPETITION.
```

Figure 3-15

---

*Contradiction as defined here is similar to Flowers (1982) notion of inferential contradiction. Stallman and Sussman (1979) also used the notion of contradiction to detect bad inferences in the analysis of electronic circuits. In their domain, electronic laws provide a much more rigorous means of setting up constraints than is possible in less orderly domains like disputes.

We call the third type of elaboration error *undetected errors*. These are the most insidious errors. They are elaboration inferences which are consistent with the problem solver's knowledge, but are incorrect with respect to the external real problem. Often these errors occur as a result of default reasoning and generally result in a planning failure. Orange-Dispute-0 illustrates this type of elaboration error. As you recall from above, the mother-mediator inferred that the sisters wanted to eat the orange. This goal elaboration is consistent with the normal use of an orange in a physical dispute context. However, one sister did not want to eat the orange, so the mother-mediator's planning failed because of this undetected error. Recognition and recovery from this type of error can be done only after later feedback. It is discussed in chapter five.

3.4.4 Goal relationship elaboration

Since each disputant in a dispute has a separate goal, the interaction of these goals provides an important characterization of the dispute which needs to be inferred. We have adopted Wilensky's (1983) description of these goal relationships as competitive or concordant. Competitive goal relationships occur when the goals of the disputants are incompatible. For example, the boys in Candy-Dispute-0 both want to eat the whole candy bar. Candy-Dispute-0, thus, has a competitive goal relationship. Concordant goal relationships describe those situations where the disputants goals are compatible and are non-interfering. For example, the sisters in Orange-Dispute-0 really have concordant goals, since one can have the fruit and the other the peel. The problem for the mediator is how to recognize and properly elaborate the goal relationship of a dispute, so that planning can take advantage of this knowledge.

Goal relationship recognition might seem at first to be quite simple, given the two example disputes in the preceding paragraph. After all, in Candy-Dispute-0 the boys both had the same designs on the same disputed object, while in Orange-Dispute-0, the sisters had different intentions on different parts of the disputed object. Thus, the obvious scheme would appear to involve matching the goals of the disputants and the disputed object. If the goals are the same type and involve the same object, then we could conclude that the dispute was competitive. This simple recognition scheme, however, has difficulty with competitive cases such as the following:

ANTARCTIC DISPUTE

> Fourteen nations are meeting to decide the fate of Antarctica's natural resources. One coalition is interested in extracting Antarctica's resources as a means of providing income for poorer nations. Another coalition wants Antarctica protected as a natural wilderness and object of scientific investigation.

This case illustrates the point that when the disputants have different goals, we cannot always infer that the dispute is concordant. Since one group of nations intends to extract the natural resources from Antarctica, while the other group is interested in preserving its current undeveloped status, the simple matching of goals would fail in this case, resulting in it being characterized as concordant. Clearly, the case is competitive. One group cannot extract natural resources without violating the other group's desire to maintain its natural state. Thus, even though their ultimate goals are different the instrumental supporting goals conflict. Specifically, physical control of the Antarctica by any group threatens the preservation of its natural state. It is from this threat to the preservationist goal that the competition is derived.

As a result of the above observation, the goal recognition strategy below includes an analysis of the supporting instrumental goals of one disputant in relation to the supporting instrumental goals of the other disputant. If at least one of these supporting goals threatens one of the other disputant's goals then the dispute is inferred to be competitive (see section 2.3.2 for the explanation of support sets and threat sets). Note also that this strategy is dependent on previous dispute classification and goal elaboration processes:

GOAL RELATIONSHIP RECOGNITION STRATEGY

1. Using the details already provided by problem classification and goal elaboration processes, first insure that the disputants all have instantiated goals.

2. If the disputants both have the same ultimate intentions on the same disputed object, then infer the goal relationship is competition.

3. If the disputants have different intentions on the same disputed object and at least one of the set of supporting instrumental goals (i.e., the "support set") of one disputant threatens a supporting instrumental goal of another disputant, then infer the goal relationship to be competition.

4. Otherwise, infer that the goal relationship is concordant.

Figure 3-16

3.5 Some implications

One implication from this model of problem understanding for case-based reasoning is that there must be memory structures that organize cases according to the basic component features of problems. This is required so that similar cases can be recalled based only on the basic features present in the initial problem representation. In the dispute mediation domain, this means we need memory categories that organize cases according to disputants, disputant goals, disputant arguments, and disputed objects. Without such memory structures, the initial bottom-up reminding of previous cases would not be possible. A full discussion of the necessary memory structure will be presented in chapter six.

Case-based reasoning supports problem understanding decisions by analogy to corresponding decisions in a similar case. Thus, when the problem solver needs to classify the problem, a recalled similar case provides a plausible suggestion for classification. When the problem solver needs to infer missing portions of the representation, a recalled similar case provides plausible elaborations. In each situation, case-based reasoning is invoked by the problem solver's need to make a decision. For this reason, we say case-based reasoning is "demand driven" (Schank and Birnbaum, 1980; Granger et al., 1984). Transfer of information between a recalled problem and a new case is constrained by domain-specific consistency and contradiction-detection knowledge. This knowledge is used to constrain not only the transfer of information between cases, but also the elaboration processes in general.

One rather obvious point that should be remembered for later chapters is that successful problem solving often depends on the correct understanding of the problem (Bobrow, 1968; Hayes and Simon, 1979; Polya, 1945). This relationship between problem understanding and successful problem solving will become more evident in the next chapter which discusses planning, the next phase in our model of problem solving. Plans are selected based on the representation of the problem (i.e., its understanding). Incorrect plan selection and application will usually lead to a failure. If a problem solver knows that misunderstanding of problems is possible, then one technique for recovering from failure is to introspectively analyze decisions made during understanding to identify potential sources of error. This will be discussed in the later chapter on failure recovery.

3.6 Summary

In this chapter, we have presented a process model of problem understanding. Problem understanding is a constructive process responsible for creating an internal representation for an external problem description. We have made four key points. First, problem understanding is composed of three specific stages: initial problem representation, problem classification and problem elaboration. Second, we have specified techniques for transferring the classification, goals, and themes of previous cases into the latter two of these stages as an improved heuristic method of problem solving based on previous experience.

Third, we recognized the risk of erroneous elaboration transfers and indicated how they can be detected and avoided by the explicit use of domain specific consistency and contradiction constraints. Finally, we have indicated the need for an overall method of default reasoning in the absence of specific cases. For classification in the dispute mediation domain, we use a simple heuristic that is biased toward the "physical dispute" classification. For goal elaboration, we use direct, thematic, and default elaboration heuristics. Sources of knowledge for these heuristics include the disputant's argument, a recalled similar case, and the disputed object.

CHAPTER IV

CASE-BASED REASONING IN PLANNING

Second. You should obtain eventually a *plan* of the solution.
(Polya, 1945)

4.1 Introduction

ORANGE DISPUTE-0

A mother arrives home from the library and finds her daughters quarreling over an orange. Recognizing the obvious similarity between this situation and her recent experience with the little boys, she suggests that her daughters stop quarreling and divide the orange equally between themselves by having the first daughter cut the orange into two pieces and letting the second daughter choose her piece first. The girls agree to her suggestion. The first daughter peels her half orange and eats the fruit. But her sister peels her half, throws the fruit in the trash, and uses the peel to bake a cake.

Planning, in general, is a process of choosing actions, collectively known as a plan, that a problem solver believes will lead to problem resolution. In Orange-Dispute-0 above, the mother-mediator selected the common sense mediation plan that we call "one cuts, the other chooses" as her suggested plan for resolving her daughters' dispute. She selected this plan, after being reminded of a previous similar case in which the same plan had proven successful. As described, the case illustrates one of the ways that case-based reasoning can support the planning process: a particular plan can be selected for investigation or employment based on its use in a previous similar situation.

Our approach to planning is based on the assumption that in many situations, a plan to solve a new problem can be generated by recalling previously successful plans for similar problems and adapting them to the current situation. This use of analogy is, in fact, common in the planning people do (Carbonell, 1983a; Luchins, 1942; Gick and Holyoak, 1980; Polya, 1945; Reed and Johnson, 1977; Rumelhart and Abrahamson, 1973; Sternberg, 1977).

In this chapter, we will present a model of planning that incorporates the use of previous experience. In particular, we will describe case-based processes that:

1. choose an overall planning policy,
2. suggest plans that should be adopted because of previous success,
3. discourage selection of plans that had failed in similar situations,
4. suggest component refinements for proper plan instantiation, and
5. predict the consequences of plan application in particular situations.

Each task mentioned above is a component of an overall planning process based on the successive refinement and instantiation of known abstract plans (Friedland, 1979; Wilensky, 1978; Wilensky, 1983). This is the default planning process that we have chosen to augment with case-based reasoning. We elaborate the specifics of our particular approach in the next section. After this overview of planning, later sections discuss the details of each planning phase and show how case-based reasoning supports each of the above planning tasks.

4.2 An overview of planning

Planning and problem solving are often used synonymously (e.g., Carbonell, 1983b; Cohen and Feigenbaum, 1982; Newell and Simon, 1972). We, however, will differentiate planning from problem solving in the following way: planning is that part of problem solving responsible for determining the goal-directed actions of the problem solver. Thus, planning is an important, but subordinate, part of problem solving. It is the stage of problem solving, in our model, that follows problem understanding and precedes evaluation of feedback and possible recovery from failures. Figure 4-1 below highlights where the planning process, which includes such tasks as solution generation and consequence prediction, fits into our overall model of problem solving.

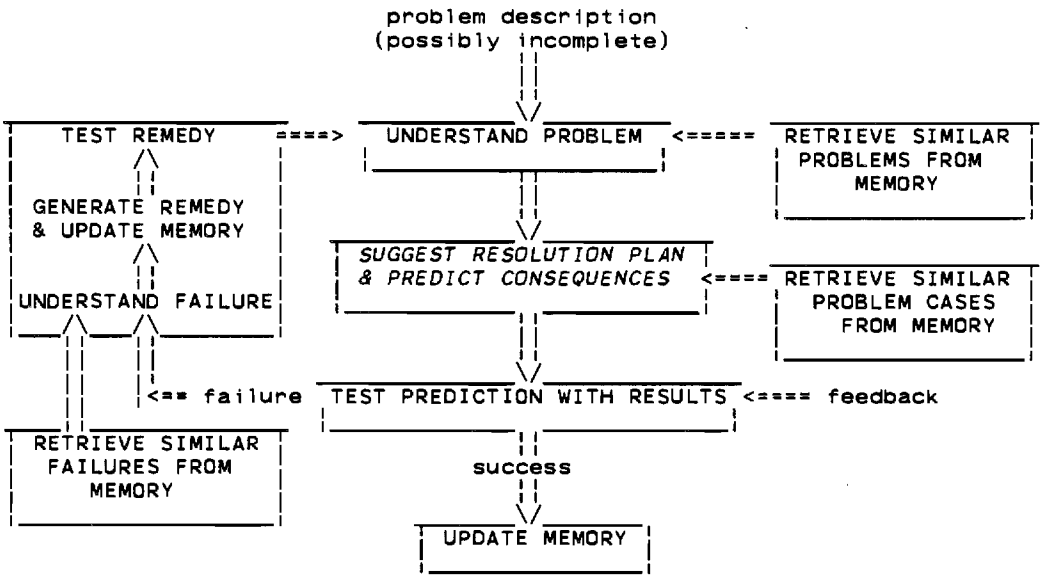


Figure 4-1

4.2.1 The overall planning process

The planning process in our model is plan instantiation (Friedland, 1979; Wilensky, 1983). Plan instantiation is a type of planning where, instead of constructing every plan from scratch for each problem, plans are selected from a set of already known abstract plan types. Beginning at the highest level of abstraction, the most promising general plan is selected and then successively refined until fully instantiated.

In our version of plan instantiation, there are four stages of selection and refinement: (1) a meta-planning process first establishes an overall planning policy which guides later planning decisions, (2) a plan selection process, beginning at the highest level of abstraction, next chooses the most promising general plan believed applicable for the problem, (3) a refinement process then specializes the general plan to the point of instantiation for the particular problem, and (4) a prediction process generates a specific set of expectations based on the assumption that all actions are executed as planned. Planning involves making hard decisions, often with incomplete information, in each of these four stages. It ultimately results in both a proposed plan of action which can be executed by some agent, and a set of expectations, which must be confirmed. Our overall model of planning is reflected in Figure 4-2 below:

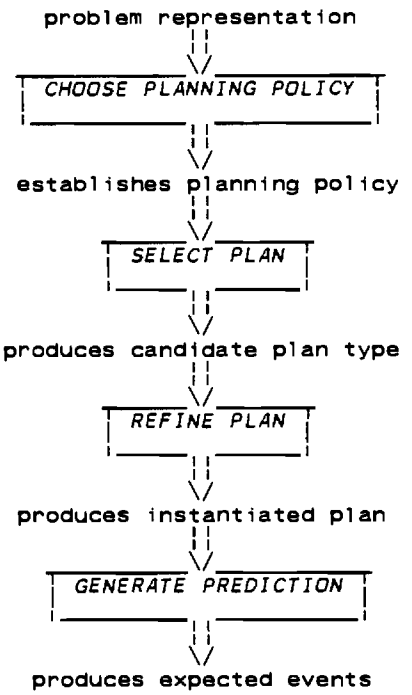


Figure 4-2

The first stage of planning is a "meta-planning" process that decides the overall planning policy under which the planner will operate. Examples of planning policies include global constraints (e.g., "only use 5 seconds of processing time to find a solution"), guidelines (e.g., "try a problem decomposition approach"), and desirable features of the evolving plan (e.g., "minimize the cost of the solution"). In the dispute mediation domain, one planning guideline is the mediator's policy of choosing "compromise" mediation plans over "all or nothing" mediation plans for competitive disputes (see chapter two, section 2.2.1). This guideline is considered a part of "meta-planning" because it involves a decision made by the problem solver about the planning process itself independent of any particular problem.

The second stage is a process of abstract plan selection. Plan selection depends on what plans are available and what process a planner uses to evaluate known plans for the current problem. The possible plans in a planner's repertoire is one part of a planner's knowledge. These plans can be specified on many levels of detail, from sequences of primitive (nondecomposable) actions to more complex abstract plans involving generalized actions (e.g., Fikes, et al., 1972; Friedland, 1979; Hayes-Roth and Hayes-Roth, 1979; Sacerdoti, 1977; Wilkins, 1984). At the highest level of abstraction, we assume there will be a small set of fundamentally different plans known to a planner. For example in the dispute mediation domain, we have identified seven general plans (see chapter two, section 2.4). In essence, this stage of planning involves selecting one of these abstract plans for further investigation.

The third stage of planning, plan refinement, is the process of selecting and instantiating an appropriate specification of the plan chosen in step 2. Whereas the previous stage decided the abstract nature of the ultimate plan, this stage is responsible for its instantiation for the case at hand. There are two parts to this refinement process. The first part is the further specialization of the plan type to the lowest possible level in the abstract plan hierarchy. The second part is the instantiation of the actions and variables in the plan using the terms at the lowest level in the semantic language used to describe the domain* and the binding of roles in the abstract plan.

---

*This is sometimes referred to as the "instance level" language, as contrasted with the "generalization" language used in a particular domain. See Mitchell (1981) for more details.

The first part is illustrated by the specialization of the abstract plan "divide equally" into the more concrete plan "one cuts, the other chooses." The second part includes deciding who will play the role of the cutter and how the "cutting" action will be executed. These concepts must also be part of the planner's knowledge and it is in this sense dependent on the semantic language used to model the domain.

Prediction generation is the final stage of our planning model. It is a forward inference process that uses prior inputs and knowledge to predict subsequent events. In many planning systems (e.g., Newell and Simon, 1972; Stefik, 1981; Wilkins, 1984), the planner is finished when a plan has been instantiated. Integrating the planner as one component in an overall problem solving system provides the opportunity for the planner to support the next phase of problem solving: confirming that the problem has indeed been solved. In order to make this later decision, a problem solver must have some means of comparing the actual results of plan execution against a set of predictions. This capability forces a planner to continue past the plan instantiation phase to produce a set of predicted actions. In Orange-Dispute-0 for example, the mother expects to see each daughter eating her half of the orange. The existence of this prediction allows her to recognize the failure of the "divide equally" plan in the instance when one daughter uses the peel to bake a cake.

4.2.2 Case-based support for planning

Case-based reasoning provides heuristic support for all the planning decisions mentioned above. The recalled decisions made by a planner in a similar planning situation provide one source of advice that influences those same decisions in the current planning problem.

A recalled similar case can support a planner during the policy making stage of planning (step 1) by suggesting the adoption of policy guidelines successfully used in a previous case. In the dispute mediation domain, for example, if the "compromise" planning policy was used in a similar case with success, then it can be transferred from that case as long as it is consistent with other planning constraints.

Recalled cases are used to suggest plans to adopt as well as plans to avoid during the abstract plan selection phase of planning (step 2). A previous similar case which included the successful use of a plan encourages the adoption of the same plan again. Conversely, a recalled case which included the unsuccessful use of a plan should discourage the planner from once more using that same plan for a similar problem. For example, if a mediator recalls that the "divide equally" plan was successful for a similar dispute over food, then that same plan can be suggested for a new case. If, on the other hand, the mediator recalls that the "divide equally" plan was unsuccessful in a similar case where the goal relationship was "concordant," then that plan can be avoided.

During the plan refinement stage of planning (step 3), a recalled case can identify a specific version of an abstract plan and suggest how the roles in the plan should be bound for the current problem. When a specialization of a known abstract plan is transferred from a recalled case, it provides an opportunity to avoid the previous plan selection phase altogether. This can happen, for example, if the recalled case employed a specific plan like "one cuts, the other chooses", then that plan may be adopted directly for the current case without first requiring the selection of the more abstract "divide equally" plan.

A recalled case can support the prediction generation phase of planning (step 4) by suggesting actions that are similar to those that occurred in previous uses of the plan. If a mediator recalled, for example, that one other time when the "one cuts, the other chooses" plan was used to resolve a dispute over food, the disputants each ate their half of the food, then a similar prediction is reasonable for another dispute over food.

Figure 4-3 summarizes these four heuristic uses of case-based reasoning to support the planning process.

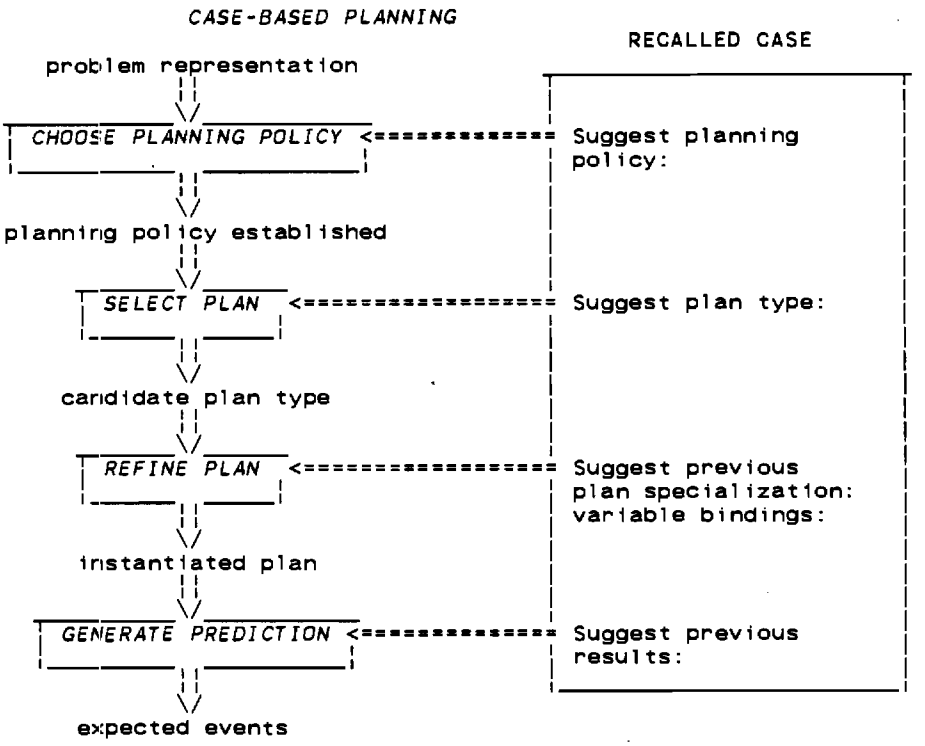


Figure 4-3

Case-based reasoning in support of planning, as in other problem solving tasks, requires that the problem solver (planner) record the problem context, the decision(s) made, and the rationale for the choice. This was the case in making problem understanding decisions and is just as important to the use of case-based reasoning for making planning decisions. Each of the above stages of planning will be discussed in more detail in later sections. Our overall planning process will be illustrated by showing the successive development of a mediation plan for Orange-Dispute-0. This will include examples of the decisions made at different planning stages as well as illustrative case-based algorithms used to support those specific decisions. These algorithms have the same basic underlying form presented in the previous chapter: retrieve a similar case, examine the decision(s) made in that case and determine their acceptability to the current problem, transfer acceptable decisions, and use default reasoning to make other decisions.

4.3 Case-based reasoning in choosing a planning policy

Planning policy decisions are decisions about planning (i.e., meta-planning). Planning policy choices direct subsequent planning actions and determine the character of the eventual plan as well as the efficiency of the planning effort. In general, there are many policies, guidelines, and criteria necessary to control the planning process. To illustrate this part of the process, we will focus on one particular planning guideline we refer to as the "compromise planning policy" (see section 2.2.1). This planning policy directs the planner to investigate plans that result in compromise solutions before considering those that result in "all or nothing" solutions in the absence of specific knowledge on which to make this decision (see Dyer (1983), Hayes-Roth and Hayes-Roth (1979), Stefik (1981), and Wilensky (1983) for other examples).

We have included a meta-planning process within our model of planning because, like most recent theories of planning in AI (Hayes-Roth and Hayes-Roth, 1979; Stefik, 1981; Wilensky, 1983), we recognize that meta-planning decisions are extremely important parts of the planning process. Separating the knowledge that a problem solver uses to make decisions about planning from specific plan instances allows us to explicitly reason about the underlying assumptions used to guide planning in similar situations. If these meta-planning decisions are excluded from the record of planning decisions, then a remembered solution, believed applicable to a similar situation, may actually fail because of a difference in planning assumptions and policies. Meta-planning decisions are both domain and model dependent. They are <u>domain dependent</u> because they are useful only when they can be made operational for a specific domain (Wilkins, 1984). For example, the "compromise" concept in dispute mediation, which is the heuristic equivalent of partial goal satisfaction, is not selected as an active planning guideline in most domains until all other options are exhausted (Wilensky, 1983). In the dispute mediation domain, however, partial goal satisfaction is

normally the most successful (i.e., default) planning guideline. Meta-planning decisions are model dependent because the decision is presumed to be important to the underlying planning process. For example, in our planning approach the "compromise" planning policy is used to guide the planner in making plan selection decisions in the absence of specific knowledge. If we had modelled this process in a classic search paradigm, we would have to insure that this control decision was used in some way to guide search. In some approaches, this is accomplished by using agendas or meta-interpreters (Stefik, 1981).*

We will address three issues in this section. First, how does case-based reasoning support a planner's policy decisions? We will provide a case-based algorithm that describes a mechanism for allowing previous cases to be used in making planning policy decisions. Second, what constrains the transfer of planning policy from previous cases? We shall see that planning policy transfers are controlled by specific policy consistency constraints. Finally, what does a planner do in the absence of useful cases? We will illustrate default policy decisions by examining the MEDIATOR's planning policy algorithm.

4.3.1 Case-based selection of a planning policy

From the perspective of case-based reasoning, a planning policy decision is pretty much like other problem solving decisions. There is a known set of alternative decisions and a rationale, consistent with the problem context, for choosing one of the decisions. A case-based algorithm for chosing a planning policy is presented below in Figure 4-4.

CASE-BASED SELECTION OF A PLANNING POLICY

1. If a previous case is already known, then using previous case go to step 2, otherwise recall a similar previous case and go to 2.

2. If the planning policy decision made in the recalled case is consistent with what is known about the current case, then transfer the planning policy and adopt it for the new case.

3. Otherwise, choose planning policy by default reasoning.

Figure 4-4

Consider how this algorithm could be used in the resolution of Orange-Dispute-0. Suppose that the problem solver was reminded of Candy-Dispute-0, which was resolved using a "compromise planning policy." Compromise is not inconsistent with Orange-Dispute-0 because there is no explicit objection to this approach in the problem presentation. Since it is not inconsistent, that same planning policy is transferred, resulting in the change in representation illustrated in Figure 4-5.

---

*One heuristic way to recognize model dependent meta-planning decisions is that their inclusion usually results in fundamental changes to representations. For example, if we represent plans as a simple list of actions, then inferring that a plan was a "compromise" plan would be made extremely difficult. In contrast, representing plans as structured objects, we can attach descriptive features to abstract plan types and decisions such as this can be made as straightforward as a table lookup.

AN EXAMPLE OF CHOOSING PLANNING POLICY

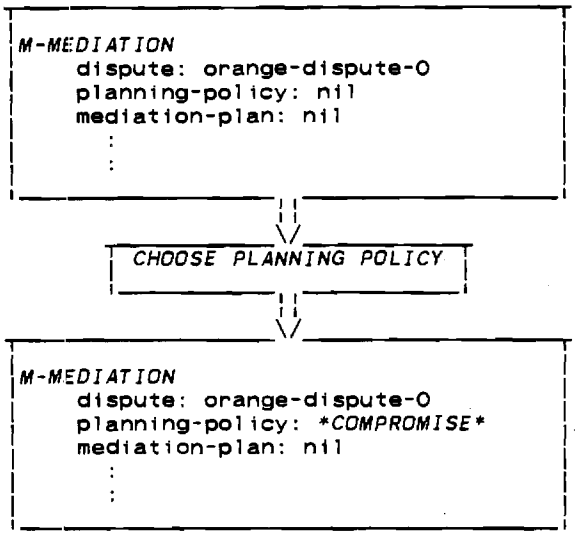


Figure 4-5

The algorithm presented in Figure 4-4 is the same as case-based algorithms presented elsewhere in this manuscript, with its emphasis in step 2 on choice of planning policy. The choice itself depends on consistency (or lack of inconsistency) judgements that constrain the transfer of policy from one case to another.

4.3.2 Constraining planning policy transfer

Planning policy decisions, like other problem solving decisions, must remain consistent with any known dependency constraints in the problem domain. For example, the MEDIATOR's "compromise planning policy" is useful only if the disputants are known or believed to be willing to consider compromise solutions to their dispute. If the disputants explicitly tell a mediator that compromise solutions are not desired, this should prevent the transfer of a "compromise planning policy" from a recalled case. This would explain the mother-mediator's planning behavior in the following version of the candy dispute:

CANDY DISPUTE-3

> quarreling over a candy bar. She overhears the first little boy shout, "I want it." To which the second boy responds, "So what, I want it too." Unable to resist the opportunity to play mediator, the mother offers to help the boys settle their disagreement. The boys reluctantly agree, but with the provision that, as the first boy says, "I don't have to share it with him." With this constraint, the mother thinks for a minute then suggests that the boys flip a coin to see who gets the candy. The boys agree and the mother continues homeward.

In Candy-Dispute-3, the boys told the mother-mediator not to suggest compromise mediation plans. This type of explicit information allows a mediator to make planning policy decisions with greater certainty. In this case, the planning policy should be "all or nothing." This planning guideline contained explicitly within the problem description effectively constrained the mother's planning decisions. This same explicit information should also prevent the transfer of an inappropriate planning policy from a recalled case. These considerations are reflected in the MEDIATOR's planning policy algorithm shown in Figure 4-6. In this algorithm, the transfer of planning policy from a recalled case is constrained by the explicit evidence provided in the problem description that is known to conflict with a possible policy transfer.

MEDIATOR'S ALGORITHM FOR CHOOSING A PLANNING POLICY

1. If a previous mediation is already known, then using previous case go to step 2, otherwise recall a similar previous mediation case and go to 2.

2. If the planning policy in the recalled case is "COMPROMISE", then check the current disputants' arguments to ensure there is no explicit opposition to a "COMPROMISE" solution and transfer the planning policy for the new case.

3. If the planning policy in the recalled case is "ALL-OR-NOTHING," then check the current disputants' arguments to ensure there is explicit desire for an "ALL-OR-NOTHING" solution and transfer the planning policy for the new case.

4. Otherwise, choose the "COMPROMISE" planning policy.

Figure 4-6

Notice that for the common sense mediation of disputes, the MEDIATOR is biased toward the "compromise planning policy." This is the default planning policy and it is changed only when there is evidence, obtained from the disputants' arguments, that it is inappropriate.

4.4 Case-based reasoning in plan selection

After a planning policy is chosen, an abstract plan is selected. Our plan selection mechanism is a plan instantiation process that does a best first selection of an abstract plan from a set of known alternatives. Based on this decision, later stages of planning successively refine the abstract plan until it is fully instantiated. Case-based reasoning helps a planner select a plan by suggesting the plan used in a similar case. This suggestion offers the possibility of avoiding the successive levels of reasoning that would otherwise be necessary to make this decision by static default reasoning. For example, if a mediator is reminded of a case which was resolved by the "one cuts, the other chooses" plan, then that specific plan can be investigated without considering the more general plan "divide equally," which can be done by several methods. In an extensive plan environment, this shortcut could represent a sizable advantage.

4.4.1 Selecting a general plan

In order to explain plan selection, we need to examine the planner's knowledge of plan alternatives and the method used to choose among these alternatives. Because our planning approach is based on the notion of stepwise refinement and instantiation, one way of looking at the abstract plan selection decision made during this stage of planning is as a first level specialization of the abstract "plan" concept. To make this notion operational, we have organized the MEDIATOR's plan knowledge in a standard abstraction hierarchy. Each plan represents a specialization of the "mediation plan" concept. This organization is illustrated in Figure 4-7:

A PORTION OF THE MEDIATION PLAN ABSTRACTION HIERARCHY

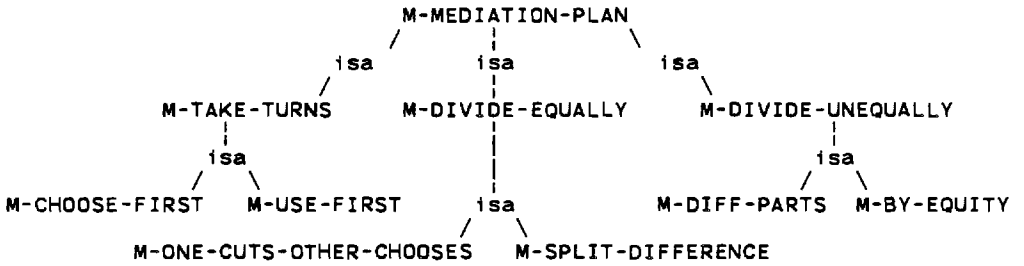


Figure 4-7

Using the semantic knowledge provided by this relationship among plans, the plan selection process is equivalent to successively refining the concept "mediation plan" to one of its next lower level plans. Using Figure 4-7, this means picking one plan from the set of three plans: "divide equally," "take turns," or "divide unequally." As you may recall, we explicitly provide these options as part of the information included in the representation of different

problem types in the slot called "usually-useful-plans" (see section 3.2.2). In this way, we provide the planner with a set of known alternatives from which a selection may be made. This knowledge, because it is told to the program, is somewhat analogous to the "book knowledge" provided to human apprentices during training (Kolodner, 1983). Besides representing the planner's knowledge of possible actions, this semantic knowledge also describes the organization used by the planner in acquiring and relating experiential (i.e., episodic) knowledge about how each plan has been used in the past. This will be discussed more in chapter six.

Given a set of alternative actions, a critical part of plan selection is the process of deciding among known alternatives (for example, choosing among the three above). In many AI planning systems, these decisions are made by evaluating the possibilities using a single static evaluation function (e.g., Samuels, 1963). Instead of using a single evaluation function applied globally to each plan alternative, we associate a set of preconditions with each plan. These preconditions, in their simplest role, provide an evaluation of plan applicability in terms of its acceptability for the problem. For example, the preconditions for the "divide equally" mediation plan are repeated below from section 2.4.1.

1. the mediator has a compromise planning policy,
2. the dispute has a competitive goal relationship,
3. the disputed object is splittable, and
4. the disputed object is not sharable.

In general, plan preconditions are made up of two different types of statements: necessary conditions for the plan's employment and exclusionary conditions which prevent the plan's further consideration. The first three statements above, for example, illustrate problem features or states that are believed necessary for successful plan application. As such, the satisfaction of these conditions can be viewed as positive evidence in support of plan selection. The fourth statement above illustrates exclusionary preconditions. It is also important to plan selection, but this type of precondition identifies conditions which are used to prevent the plan's selection. This represents evidence in opposition to the plan's use for a problem featuring this condition (e.g., the "sharable" feature in the example above). Preconditions thus contain both types of statements: those which indicate support for the plan and those which provide criteria to exclude the plan's selection.

Preconditions mean different things in different planning systems. As described above, we define preconditions as states over which the planner has little or no control. This is reasonable in the dispute mediation domain or other planning situations where the planner is functioning in an advisory role. Mediators, for example, are third parties to disputes and usually do not try to alter the dispute situation. Other planning systems (e.g., Newell and Simon, 1972; Sacerdoti, 1977; Wilensky, 1983) use the term preconditions to mean "subgoals." In these planners when the preconditions for a plan are not true, a subgoal is created to attempt to satisfy the preconditions.*

---

*The meaning of preconditions as used by "means-ends analysis" planners corresponds to what Schank and Abelson (1977) refer to as "controllable preconditions" and "mediating preconditions". Our meaning of preconditions matches Schank and Abelson's definition of "uncontrollable preconditions."

In the simplest situation, preconditions are conjunctions of boolean tests whose results are certain. This is the case, for example, in the preconditions shown above for the "divide equally" plan. Far more difficult plan selection decisions are necessary under conditions of uncertainty. Suppose, for example, that preconditions are not completely satisfied for any of the possible plan alternatives. Selecting a plan under various levels of uncertainty is an important issue in planning. Although we do not directly address this issue, case-based reasoning (i.e., the transfer of a plan selected in a similar situation which proved successful) seems to be a promising means of dealing with plan selection under uncertainty.

Using the preconditions defined for each plan alternative, one way an abstract plan can be selected is by accepting the first plan whose preconditions hold for the given problem. For Orange-Dispute-0, which we are using to illustrate the planning process, this results in the mediator selecting the "divide equally" mediation plan because it is the first plan evaluated whose preconditions are all satisfied. This decision is recorded in the mediation case frame by specifying a filler for the "mediation-plan" slot as indicated in Figure 4-8:

AN EXAMPLE OF SELECTING AN ABSTRACT PLAN

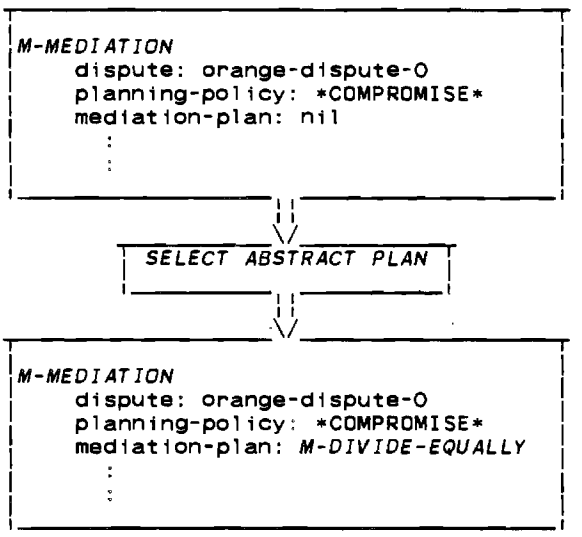


Figure 4-8

4.4.2 An algorithm for case-based plan selection

As with previous case-based algorithms, a problem solver first attempts to transfer the decision (i.e., the selected plan type) from a previous case within known domain constraints (i.e., the plan's preconditions) and only if that is unsuccessful resorts to a static line of default reasoning. Otherwise, the cased-based algorithm shown in Figure 4-9, for case-based plan selection, is similar to those presented earlier.

A CASE-BASED ALGORITHM FOR PLAN SELECTION

1. If a previous case is already known, then using that case go to step 2, otherwise recall a previous similar case and go to step 2.

2. If the preconditions for the plan used in the recalled case are satisfied, then transfer the plan type for the current case.

3. Otherwise, select plan by default reasoning.

Figure 4-9

In the following fragment from the MEDIATOR program, we see how this algorithm applies to the selection of a plan for Orange-Dispute-0. In this situation, the MEDIATOR has already retrieved Candy-Dispute-0 as the most similar case, so step 1 of the algorithm does not apply. In step 2, the plan used in Candy-Dispute-0 is identified and its preconditions tested. Since

the plan's preconditions are found to hold in the current case, the plan is transferred and applied to Orange-Dispute-0.

I/O BEHAVIOR OF THE MEDIATOR DURING CASE-BASED PLAN SELECTION

```
ATTEMPTING TO SELECT A NEGOTIATION PLAN TO RESOLVE THE DISPUTE
   IDENTIFIED AS #<M-PHYS-DISPUTE 22475211> (orange dispute-0).

Using previously recalled case,
   where two children are quarreling over a candy bar.
   It was resolved using the plan known as "one cuts the other chooses."
Checking for applicability of that plan to the current case..

My reasoning is as follows:
   It normally doesn't make sense to share ORANGE1,
   since its functionality is destroyed by its consumption,
   but it can be divided without loss of functionality;
   when this is considered with a compromise planning policy
   and my inference that the parties' goals are in competition;
   all indicate that "one cuts the other chooses" is a reasonable plan.
Selecting the plan "one cuts the other chooses"
   for this dispute and instantiating.
```

Figure 4-10

As illustrated in Figure 4-10 , when the plan used successfully in a previous case is identified for possible transfer to the current case, its preconditions are evaluated. If the plan's preconditions are satisfied, then that plan is selected for the current case. This avoids the possibly lengthy evaluation of other alternatives and results in planning behavior that is biased by previous successful planning experiences.

4.4.3 Case-based explanation

Figure 4-10 also illustrates how the program explains its reasoning by citing the known preconditions for the selected plan. The capability of a problem solver to explain its reasoning is very important for insuring confidence in the program's behavior. Explanation is a complex process that requires knowledge about the problem solving process as well as the problem domain. One of the general situations requiring explanation is when the problem solver needs to tell what data and inference were used to reach a decision. Since the MEDIATOR's beliefs about the nature of the problem are the results of previous reasoning steps (during the understanding stage), explanation such as produced above is equivalent to those produced by problem solving systems using a rule-based paradigm (e.g., Davis and Lenat, 1980). If we had formulated the MEDIATOR in terms of that paradigm, the explanation would represent the recapitulation of the rules that were responsible for the decision. As currently implemented, explanation is accomplished via explicit knowledge of the preconditions for each plan. With the additional information provided by a recalled case, the opportunity is available to expand the explanation capabilities of a problem solver by use of an explanation based on analogy. Except for the dialogue produced by the MEDIATOR during problem solving, we have not explored the possibility of constructing explanations by explicitly pointing out to a client the analogy between his case and a previous case. This is also a possible avenue of future research.

4.4.4 When a plan cannot be transferred

Just because a plan was useful in a similar case does not mean it will always be applicable to the current problem. After identifying the plan used in an analogous situation, a problem solver needs to prevent the transfer of an incorrect plan. In general, domain specific constraints are used to perform this check on transfer in case-based reasoning. Plan preconditions perform this duty for case-based reasoning in the plan selection process because they constrain the plan's transfer in the same fashion as they prevent the selection of an inappropriate plan during default plan selection. When a plan is identified for possible transfer from a recalled case, the preconditions for that plan are evaluated. If that evaluation fails, the transfer is aborted and default reasoning is begun.

Default reasoning in plan selection, like other static means of decision making, can employ any standard technique that is reasonable for the problem and the domain. The default plan selection process used by the MEDIATOR is a simple process of choosing the first plan whose preconditions prove applicable for the current case. This algorithm is presented in Figure 4-11 below:

MEDIATOR'S BEST FIRST ALGORITHM FOR DEFAULT PLAN SELECTION

For each plan type in the set resulting from the union of plan types retrieved from the "usually-useful-plans" and "other-plans" slots in the dispute representation

> Test the applicability of each plan's preconditions against the known conditions in the current dispute. If a plan's preconditions are appropriate for the current dispute, return that plan type.

When no more plan types are available, signal an error.

Figure 4-11

4.5 Case-based reasoning in plan refinement

Plan refinement, the third stage of our planning model, involves further specialization and instantiation of an abstract plan chosen in stage two and believed appropriate for the current problem. Whereas stage two decided the abstract nature of the ultimate plan, this stage is responsible for its instantiation for the case at hand. The decisions made during this phase are of two basic types. The first type are specialization decisions that "push" the abstract plan to the lowest (i.e., most specific) possible level in the plan generalization hierarchy (shown in Figure 4-7). Decisions of the second type include the instantiation of specific actions and the binding of roles in the evolving plan.

Both these processes can use the heuristic advice provided from a recalled case. Plan specialization, for example, can be avoided completely, if the recalled case is already specialized to the lowest level in the abstract plan hierarchy. Plan instantiating can be aided by using the previous bindings as guides. For example, if the recalled case assigned the "cutter" role in the "one cuts, the other chooses" plan to the older disputant when the disputants are of type "children," then that guidance can be considered in lieu of other constraints as a heuristic for assignment of the "cutter" role in the current instantiation.

4.5.1 Refining a plan

Further specialization of the abstract plan is a continuation of the plan selection process discussed in section 4.4. This step requires knowledge of plan types more specialized than the current plan and a method of choosing among these alternatives. As long as there are more specific plan types lower in the abstraction hierarchy and positive precondition tests which indicate their applicability to the current problem, the planner can continue to refine the plan type. Using the abstract plan hierarchy shown earlier in Figure 4-7, for example, the MEDIATOR would consider each of the two known refinements of "divide equally" as alternate possible specializations. The preconditions for "one cuts, the other chooses" and "split the difference" will each be tested for applicability exactly as explained above. For Orange-Dispute-0, this results in the plan "one cuts, the other chooses" being selected because its preconditions are satisfied. This is represented in Figure 4-12 as a replacement of "divide equally" in the "mediation-plan" slot of the mediation case frame at the top of Figure 4-12 with "one cuts, the other chooses," as shown at the bottom.

AN EXAMPLE OF ABSTRACT PLAN REFINEMENT

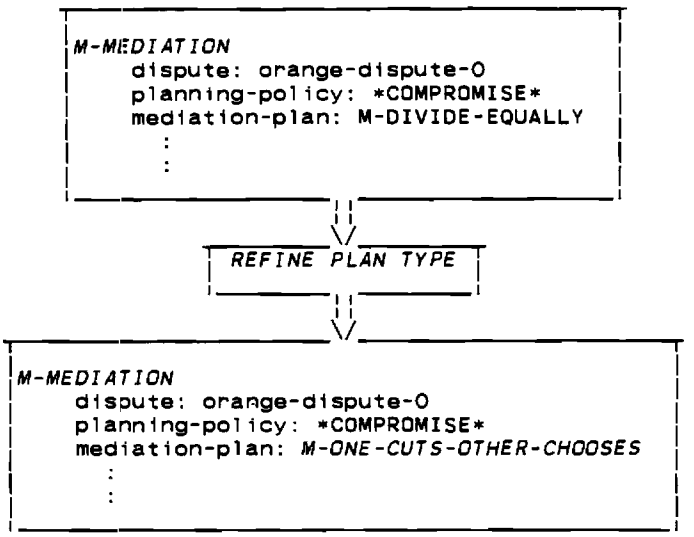


Figure 4-12

The second part of plan refinement requires binding roles, instantiating variables, and specifying procedures of the abstract plan. A planner performs these tasks by using knowledge about the options available and knowledge about the role being instantiated. For example, when a role needs to be instantiated and a planner knows that the role must be filled by a "person," the knowledge about how many "persons" have been identified allows the evaluation of alternatives for role binding. To illustrate this successive refinement process, we will present the successive changes in the procedure description for the abstract mediation plan "divide equally" as it is made more specific for Orange-Dispute-0. Our description will be in English for readability. After that, we will describe processes for doing this in detail.

Suppose after stage two, the plan chosen for resolution of Orange-Dispute-0 is "divide equally." The procedural description for this plan can be seen in Figure 4-13.

ABSTRACT PROCEDURE DESCRIPTION FOR "DIVIDE EQUALLY"

1. Divide the disputed object into as many equal portions as there are disputants.

2. Assign an equal portion to each disputant.

Figure 4-13

Notice how general this procedure is. There is no specification of the "divide" action, the "disputed object", nor the portions involved. More importantly, it doesn't specify who performs the divide and assignment actions. The next level of refinement involves specializing the plan to the "one cuts, the other chooses" version shown in Figure 4-14. This is done through the plan refinement process explained in the beginning of this section.

ABSTRACT PROCEDURE DESCRIPTION FOR "ONE CUTS, THE OTHER CHOOSES"

1. One disputant, the "cutter", cuts the disputed object into as many equal portions as there are disputants.

2. Each disputant, other than the "cutter", chooses a portion of the disputed object, the remaining portion belongs to the "cutter."

Figure 4-14

This refined procedure is somewhat better specified. The "divide" action has now been specialized as a "cut" action and the "assign" action has been made more specific to include restrictions. In addition, one role has been specified as the "cutter" (i.e., the actor who performs the "cutting" action). However, the procedure is not yet completely instantiated. For example, a planner still needs to decide which actor will be assigned the "cutter" role. At the lowest level of abstraction, these features will be filled in and the plan will be fully instantiated as illustrated in Figure 4-15 below:

INSTANTIATED PROCEDURE DESCRIPTION FOR "ONE CUTS, THE OTHER CHOOSES"

1. Sister1 cuts orange1 into two pieces, piece1 and piece2, using a knife.

2. Sister2 chooses either piece1 or piece2, the remainder belongs to sister1.

Figure 4-15

As a completely instantiated plan, the "cutter" role has been bound to sister1 and the "disputed object" variable has been bound to orange1. Notice that an instrument, a knife, has also been specified and instantiated as necessary to effect the cutting.*

---

*The variable binding stage of plan refinement is so "obvious" that many people take it for granted. However, when we look at the errors that people make during planning we can begin to appreciate the importance of these "obvious" steps. Consider the following example of a planning error related by Donald Norman. The hurried housewife is preparing for a dinner party. In her haste, she puts the salad in the oven and the cake in the refrigerator. According to the model of planning we are using, this error occurred not because the housewife had the wrong plan but because she made an error in "variable binding."

One way to view the plan refinement process as we have described it above is by analogy to script selection and instantiation as implemented by Cullingford (1981) in SAM. Each of the procedures described above for various levels of the "divide equally" plan could easily be considered as a kind of script. At each level of abstraction the "script" for the plan would provide the appropriate restrictions on variable binding necessary to refine the "script" from the more general sequences of stereotypical actions to more specific ones. So that events not only become more specific as shown above, but also become more finely grained so that one abstract event might map to several actions at a lower level of abstraction. Preconditions, as we have described them above, serve an analogous purpose to that provided by script "triggers" and provide a heuristic indication of plan (and by extension "script") applicability. We have not explored this apparent relationship between earlier script research and our model of problem solving, except to the extent that we believe there is a fairly direct applicability in the planning process.

4.5.2 An algorithm for case-based plan refinement

Case-based reasoning supports both of the basic plan refinement functions described above. It allows the planner to avoid the long static line of reasoning necessary to refine a general plan type down to a specific plan type and it assists the planner in deciding on the binding of variables. If all possible planning alternatives are known and hierarchically organized as part of a planner's a priori knowledge, then default plan selection and refinement processes (described earlier) can proceed in a methodical top down fashion to select and refine the chosen abstract plan. It is this top down search of the hierarchy of plans that can be avoided by making an analogy to previous cases. Instead of evaluating each of the intervening plans, for example, the MEDIATOR in Orange-Dispute-O selects the specific plan called "one cuts, the other chooses" (see Figure 4-10). A previously instantiated plan also allows the planner to use specific role bindings and other decisions made in instantiating an old plan to derive a fully instantiated new one (e.g., choosing the "cutter" or the "cutting" action of "one cuts, the other chooses") .

The integration of case-based reasoning with the default plan refinement and variable binding actions necessary during this stage of planning is reflected in the following case-based algorithm. Roughly, here's how it works: First it performs specialization. Then, when the refined plan is finally a type which cannot be further specialized, we focus on its instantiation. Using the identified plan type and the current case as a guide, a problem solver can probe his memory to "reconstruct" previous experience with the plan in similar situations. This process is a computational analog to the psychological notion of reconstructive memory (Bartlett, 1932; Kolodner, 1984; Loftus, 1978, 1979). A reconstructed experience provides a case whose components are abstractions or possibly parts of many different cases. We refer to these as "composite" cases. A composite case can be treated as if it were a real exemplar and used as a source to guide the transfer of specific parts to the current case. If the "composite" use of the plan is evaluated to be similar to the current case, then the individual components are matched to provide the final instantiation of the plan. If the reconstructed plan experience is not judged to be similar to the current case, then the plan is instantiated by default reasoning. We will expand on specific parts of the algorithm presented in Figure 4-16 later in this subsection.

A CASE-BASED ALGORITHM FOR PLAN REFINEMENT

1. If the selected plan is already a known specialization of a general plan, then go to step 2, otherwise specialize all general plans by default reasoning.

2. Recall previous similar instantiations of the selected plan.

3. If the components of the recalled plan instance are judged similar to the components of the current case, then use the previous plan to guide the binding of roles in the instantiation of the plan for the current case by matching the corresponding parts of the recalled use of the plan and the current case.

4. Otherwise, instantiate the plan components by default reasoning.

Figure 4-16

The first step of this algorithm is essentially the same process used in plan selection. This was discussed in the earlier plan selection section. The best first algorithm used to perform plan the refinement process is shown in Figure 4-17 below.

A BEST FIRST ALGORITHM FOR DEFAULT PLAN SPECIALIZATION

For each specialization plan in the set of known specializations of the given general plan,

> Test the applicability of each plan's preconditions against the conditions in the current case

Until either a plan's preconditions are appropriate for the current case and that plan is returned as the newly refined plan or there are no more known specializations and the general plan is retained as the default refined plan.

Figure 4-17

After the plan type has been specialized as far as possible, the next phase of plan refinement is plan instantiation. This process, as opposed to others in case-based reasoning that we have described, involves the use of a reconstructed component of a case instead of a specific component exemplar provided from a recalled case.* For example, in considering the plan "one cuts, the other chooses" the MEDIATOR constructs an instance of the plan which contains recalled components specifically tailored to the current case. This is used in lieu of the plan exemplar that could be provided by the previously recalled case. Thus, the reconstructed use of the plan is a composite recollection of previous uses. This reconstruction is performed by using components of the current case to probe the problem solvers memory for experience with a given plan. The current disputants, for example, are used to direct memory retrieval of plan exemplars when the disputants were of the same or similar type as the current disputants and the disputed object in the current case is used to retrieve a previous similar disputed object involved in previous exemplars of the plan, etc. This insures that the composite plan experience is as similar as possible to the current case for each of its major components. The retrieval process, which will be described in detail in chapter six, is essentially the same process used to recall similar cases. The difference is that we retrieve components themselves instead of the cases in which those components occur and we use the recalled components to "fill in" an instance of the plan type in order to construct a composite of the plan's previous use.**

---

*This was done to take advantage of the fact that our conceptual memory allows these composite components to be obtained easily and to investigate the advantage a reconstructed component might offer. Because of our limited investigation, however, we can make no claim for or against the use of composite cases at this time.

**This constructed composite in some respects corresponds to an operational definition of the psychological notion of a "prototype," as defined by Anderson (1980), Hayes-Roth and Hayes-Roth (1977), or Rosch (1977).

As a result of this reconstructive process, we have a prototypical experience of the named plan's previous use which can now be employed to guide the final instantiation of the plan. Before this can be done, we need to test the reconstructed plan instance to insure that the problem solver's previous experience with the plan is applicable to the current case. This test guards against the use of previous experience, when the new situation is novel enough to warrant the use of default reasoning and, by extension, constrains the use of case-based reasoning to those areas judged similar. This test is performed by an evaluation function which "rates" the reconstructed plan component by component to determine its similarity to the current situation. In the MEDIATOR, this evaluation function embodies the same ideas used to evaluate the similarity of recalled cases when more than one case is recalled from memory. This process is described at length in chapter six. At this point, we will only mention that each component is compared to the current case and contributes to a rating of the case's overall similarity based on a theory of the importance of different components. If the weighted score is high enough the reconstructed case is accepted. In the current implementation of the MEDIATOR, a perfect score is 17, and as long as the reconstructed previous experience scores above a 9 it is considered acceptable.

This is the reasoning used in the MEDIATOR program to instantiate the contract is expected to result following the plan's acceptance. The reconstructed exemplar of the selected plan type first results in a prototypical plan experience. Next the "contract" component of that reconstructed instance is evaluated for its similarity on the four most important components of disputes: disputant goals, dispute type, disputed object, and disputants. This evaluation effectively constrains the transfer of the previous contract experience if the judged similarity is not rated highly enough. This behavior is illustrated in the following excerpt from the program during Orange-Dispute-0. The sample in Figure 4-18 shows the program as it considers the contract used in Candy-Dispute-0. Because this is the only other experience known for the plan "one cuts, the other chooses", at this point the reconstructed contract is identical to the contract used in that case.

I/O BEHAVIOR DURING EVALUATION OF RECONSTRUCTED CONTRACT

```
I suggest that the plan called "one cuts the other chooses" be used.
INSTANTIATING EXPECTED CONTRACT RESULTING FROM PLAN APPLICATION.

ATTEMPTING TO RECALL PREVIOUS EXPERIENCE WITH ONE CUTS THE OTHER CHOOSES
  reconstructing my previous experience with this plan
     results in a contract identified as #<M-DIVIDED-OBJ-CONTRACT 40343224>
  checking the applicability of this contract to this situation ...

Evaluating old contract based on the four invariance
   features disputant goals, dispute type, disputed object, and disputants.

With a rating of 15 out of 17,
   the old contract is judged acceptable based on these criteria.
   using it to guide current contract construction ...
   matching SISTER1 with BOY1 ...
   matching SISTER2 with BOY2 ...
   matching ORANGE1 with CANDY1 ...
   matching
(*GOAL* (*INGEST* (ACTOR SISTER1) (OBJECT ORANGE1))) with
(*GOAL* (*INGEST* (ACTOR BOY1) (OBJECT (HALF CANDY1))))...
   matching
(*GOAL* (*INGEST* (ACTOR SISTER2) (OBJECT ORANGE1))) with
(*GOAL* (*INGEST* (ACTOR BOY2) (OBJECT (HALF CANDY1))))...
   transferring other components of contract unchanged.
```

Figure 4-18

In theory, the matching of components between analogues which is illustrated at the bottom of Figure 4-18 would be directed by domain knowledge. For example, we need not consider the possible match of orange1, in the current case, with boy1 in the reconstructed contract, because the problem solver knows that orange1 is the "disputed object", so the match is directed to the filler of the disputed object slot. A harder problem involves the matching of components which play different roles within the component representation. In the example above, why not match sister1 with boy2? This match depends on explicit plan constraints or criteria that can be used to identify the likely component analogues. In the plan "one cuts, the other chooses" for example, the heuristic associated with the "cutter" role suggests that if the disputants are children, then select the oldest child as the "cutter." This same heuristic in conjunction with the knowledge that boy1 is filling the "cutter" role allows us to identify the appropriate match between components of the old and new case. This process of mapping between an old case and an analogue is equivalent to that described by Burstein (1983). In practice, the current implementation uses a very simple matching process guided by the slot correspondences in the two representations. In some situations, most notably for the plan "take turns where the worst goes first," the matcher does evaluate domain specific criteria to determine which disputant is the "worst." In using Book-Dispute-0 as an analogous

case for the Condo Dispute, for example, the matcher first recognizes that "Fred and Ethel" own the smallest proportion of the condo and uses this recognition to match them with the student who had the lowest GPA who also was designated to fill the "worst" role.

When reconstructed plan components, like the contract, are judged inappropriate for transfer to the current case, the component is instantiated by default reasoning. This process is essentially the same as that employed in other planners that use the plan instantiation method (Friedland, 1979; Wilensky, 1983). In this process, domain knowledge also plays an essential role in guiding the planner to the proper instantiation. For example, in deciding the proper "cutting" action for the "one cuts, the other chooses" plan, the planner needs to reason about the disputed object. Thus, oranges or candy can be "cut" by "breaking" or "tearing" as well as literally "cutting." But, liquids such as orange juice must be "cut" by pouring into two different containers, etc.

4.6 Case-based reasoning in prediction generation

The final stage in our model of planning is the prediction of events that are believed likely to occur as a result of the instantiated plan's execution. This is necessary to provide expectations which can be used to evaluate the final results and determine success or failure of the problem solving experience. This process involves two different types of predictions. The first type is an expectation about likely events when all goes as anticipated by the planner (i.e., the results are "ok"). The other type of predictions are projections about what might happen under various types of error conditions (i.e., what if the goals were misunderstood?). These latter predictions are used to assist in the classification of failures, which is part of failure recovery. They will be discussed in chapter five. This section is concerned only with the generation of predictions that are used to confirm the success of the planner's expectations.

Case-based reasoning, in this stage of planning, can assist the planner by suggesting that events similar to those that occurred in a previous case will happen for the current one. This makes the generation of predictions more focussed and avoids an attempt to enumerate all possible alternatives.

4.6.1 Generating predicted actions

When a planner has control over all the factors in the problem, prediction generation is equivalent to generating the actions and their resultant states in the plan. When there are other actors that can affect the results (e.g., Sacerdoti, 1977), the process of prediction generation involves merging the results of plan application with the expected behavior of the other actors. In dispute mediation, for example, a mediator must account for the results of a mediation plan in conjunction with the understood goals of the disputants. The mediation plan "one cuts, the other chooses," for example, specifies that each disputant will receive an equal portion of the disputed object. The plan makes no statement about what the disputants will do with their portion. This is where the prediction generation process is required.

We will illustrate the results of prediction generation for Orange-Dispute-0 in Figure 4-19 below. This figure shows a portion of the "divided object contract," instantiated as one component of the "one cuts, the other chooses" plan. As a consequence of "one cuts, the other chooses," ORANGE1 is expected to be divided in half. This is reflected in the "divided object contract" as the fillers for the "part-a" and "part-b" slots. The disputants, sister1 and sister2, were previously inferred to have "M-INGEST" goals during problem understanding. This is reflected in the representation of the sisters' "has-goal" slots in the contract. After prediction generation, the "predictions" slot in the contract frame has been filled with an instance of the "M-POSSIBLE-EVENTS" frame. A portion of that frame is shown at the bottom of Figure 4-19.

AN EXAMPLE OF PREDICTION GENERATION

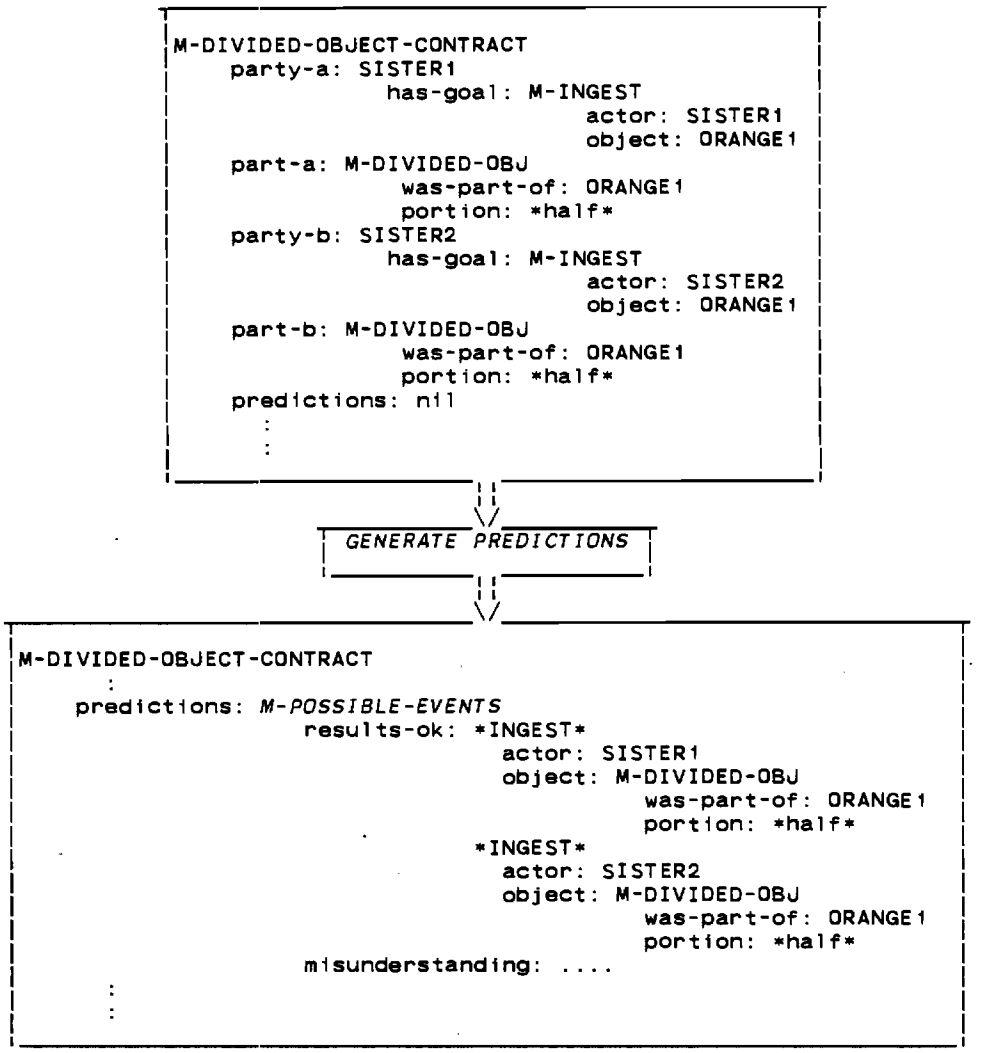


Figure 4-19

As shown above, this instance of the "divided object contract" provides the prediction that if all goes as expected (i.e., "results-ok"), then the two sisters will each consume their half orange. Predictions can also involve constructing "hypothetical" events that are indicative of certain types of errors in a problem solver's reasoning. The MEDIATOR, in the example above, also generates predictions for possible events that can be used to assist in failure classification. One possible type of failure is an understanding failure. If the sisters use their half orange for something other than an ingest action with the orange, then a misunderstanding error (i.e., the goals were inferred incorrectly) could be inferred. This would be represented in the above instance of "M-POSSIBLE-EVENTS" as the filler for the "misunderstanding" slot.* By making such predictions, a problem solver has a better chance of recognizing either success or failure. If the actual results fail to match those anticipated to occur, then a problem solver uses the other predictions (e.g., misunderstanding) to help assign blame for the error during failure recovery.

---

*This process corresponds to an aspect of planning and problem solving that is often referred to as "what-if" reasoning (Chandrasekaran, 1983) or the "generation of hypothetical cases" (Rissland, 1984).

4.6.2 An example algorithm for prediction

The case-based process for transferring previous events proceeds as in the earlier stages of planning. The results of the previous case are used to guide the instantiation of predictions for both the successful employment of the plan as well as unsuccessful uses. The recall of previous failures can be especially useful in predicting the failure portions of predictions. For example, since the MEDIATOR failed to properly understand Orange-Dispute-0, this case can be used as a negative exemplar for illustrating the events that can follow from a misunderstanding error. As in other planning stages, domain or other planning knowledge prevents the transfer of inappropriate predictions. For example, if the problem, as presented, required no elaboration, then the generation of predictions concerning understanding failures would not be necessary for the current case.

We have explored this area of planning only briefly in this current research effort. This does not mean that prediction generation is unimportant. On the contrary, as we will see in failure recovery, it is the key to assisting the problem solver in recognizing success and directing recovery during failure. In the current implementation, the possible events are hand constructed for all the cases and are thus a priori knowledge. The lone exception to this is the generation of the predictions for the situation when the plan works as expected. These predictions are governed by the prediction algorithm shown in Figure 4-20.

THE MEDIATOR'S ALGORITHM FOR DEFAULT PREDICTION GENERATION

1. If the instantiated plan is a "compromise" plan, then expect the disputants to perform actions as indicated by their "desired goal" state except that the actions will be modified according to the effects of the plan.

2. Otherwise, expect the disputants to perform their "desired goal" actions.

Figure 4-20

4.7 Some implications

In order to reason about plans as we have been describing the planning process in this chapter, we need access to a conceptual memory of plan experiences organized in terms of the plans used and their results under different problem conditions. We also need to organize failure experiences in order to make them available for retrieval and use in failure prediction. As we will see in chapter six (i.e., section 6.3.3), these are two of the different organizations of cases for which we have employed "generalized episodes" as an organizing methodology. When cases are organized in this manner, they can be retrieved and made available to the problem solver and enable case-based reasoning in support of the planning and prediction tasks specified in this chapter.

During the course of this research, the MEDIATOR program evolved as several different implementations.* The primary implementation of the MEDIATOR, used to illustrate the program's behavior in this chapter, employed a static set of preconditions that controlled the plan selection process as explained earlier. As part of a later modification, the preconditions were removed in an effort to allow the program to inductively learn the applicability conditions associated with each of its known plans. The motivation for this change was to investigate the adjustments necessary to integrate the inductive learning of control knowledge into our problem solving model. The inductive technique employed was an adaptation of the "candidate elimination" algorithm (Mitchell, 1983). As explained later in chapter six, we use a set of primitive concepts to construct a semantic model of the domain.

These concepts correspond to the "generalization language" used by Mitchell to describe a "version space" for a rule-based problem solver. We use this notion for each of the components of dispute cases to build up a version space of experience with each plan type. Different version spaces are associated with both positive and negative training instances of each plan type. Instead of having to keep these version spaces in special data structures for use by the candidate elimination algorithm, we were able to access the appropriate components needed by the algorithm directly from memory structures already provided for accumulating and generalizing cases in our conceptual long term memory.

---

*The program started out being called the NEGOTIATOR and still another version was called SOLOMON.

The modifications in the program used to investigate induction were implemented only to the extent that it confirmed our expectations that the overall model could be modified to accommodate this type of inductive learning. The fact that our conceptual memory, as designed to support case-based reasoning, could also be used to support an inductive learning process came as a pleasant bonus. There were other aspects concerning this modification to the program, however, that still need further investigation. The meaning and role of preconditions, as we defined them above, becomes less clear when a problem solver is supposed to learn plan applicability conditions inductively. For example, even if a plan failed in a recalled similar case we might still want to attempt the same plan for this case in order to confirm our previous failure. We might also want to reattempt a previously failed plan as long as there was some difference between the new case and the recalled old case. This latter investigation is necessary to refine applicability conditions for disjunctive type precondition tests (Dietterich and Michalski, 1983). In this mode of operation, the planner's preconditions become merely advisory until some state of expertise is developed. Depending on the stage of learning, preconditions can provide confident advice on the plan's applicability only when there has been previous training cases that provide complete and unambiguous version space coverage. Otherwise the problem solver is continually faced with a trial and error investigation of the version spaces associated with each plan. This makes the program more of a discoverer and less of a performer; with a corresponding effect on the efficiency of the problem solver.

In instantiating plans, we chose to use a reconstructive approach instead of transferring the component found in the most similar case. We did this because we felt this might result in the construction of a "better" component. It allowed us to tailor the current case and possibly avoid an attempt to transfer a bad component from an otherwise good case. For the most part, this was the situation. However, as more cases employed the same plans, we noticed that the composite experiences made for some unusual (i.e., nonsensical) reconstructions. For example, we might get a contract whose disputants are "polities" but whose goals were "ingest" goals. Even though the composite proved more effective overall, we still need to insure that the components are retrieved from memory within the confines of the known constraints on their overall combination in a reconstructed "prototype."

4.8 Summary

Planning is a decision making process responsible for the choice of actions that a planner believes will achieve a goal. We have presented a process model of planning based on a plan instantiation and refinement approach. This process, which shares some similarities with other hierarchical approaches to plan instantiation, is novel because it explicitly includes both meta-planning prior to domain planning and prediction generation for positive as well as negative expectations. In the mediation of disputes, for example, the mediator-planner chooses an abstract plan that he believes will lead to an acceptable resolution of the dispute within the context of some basic planning policies. This abstract plan is further refined until completely instantiated and predictions are generated as necessary. Case-based reasoning can assist this process by suggesting the corresponding decisions made in a similar case. We have provided example algorithms for each planning stage that illustrate methods of constraining inappropriate analogical transfer and use default reasoning when previous experience is judged inappropriate.

One notable aspect of case-based planning is that, unlike other planners (Sacerdoti, 1977; Stefik, 1981; Wilensky, 1983), a case-based planner does not have to first create a bad "all or nothing" plan before criticizing and constructing a satisfactory plan to achieve a "partial goal" (i.e., compromise) plan. This capability is the result of including an explicit planning policy option in our planning process.

CHAPTER V

CASE-BASED REASONING IN RECOVERY FROM FAILURE

"What is exciting is failure." (Schank, 1982)

5.1 Introduction

WINDOW DISPUTE

> Two men are quarreling in a library. One wants the window full open and the other wants it closed. The librarian, hearing the clamor, suggests they split the difference and open the window half way. Both men reject this suggestion, neither seems willing to accept a compromise solution. Finally, the librarian asks the first man why he wants the window open: "To get some fresh air." She asks the other man why he wants it closed: "To avoid the draft." After thinking a minute she opens wide a window in the next room, bringing in fresh air without a draft. The men nod their approval and quiet is restored to the library.

Failures are ubiquitous in problem solving. The window dispute is typical of many problem solving cases in that the solution was derived only after a failure had occurred. In this case, the librarian failed to immediately suggest an acceptable resolution to the quarreling men. Faced with this realization, she needed to suggest another solution. Her problem solving skill ultimately paid off in the derivation of an acceptable solution to the dispute. This points to one of our basic assumptions of problem solving which differentiates our research from most others in AI (e.g., Carbonell, 1983a; Newell and Simon, 1972; Stefik, 1981; Wilensky, 1983; Wilkins, 1984): *problem solvers must be able to deal with failures.* The story above also illustrates another general feature of problem solving: it often involves an iterative exploration of seemingly reasonable solutions to find at least one that will satisfy the goals of the problem solver. The librarian's "split the difference" suggestion was a reasonable solution, given what she knew of the men's goals. In most cases, each unsuccessful iteration provides new information allowing the refinement of a problem solver's reasoning which ultimately results in an acceptable solution. In the case of the window dispute, the librarian was able to redirect her reasoning away from a "split the difference" solution as a result of her initial failure to resolve the dispute.

This chapter is about recovery from failures in problem solving. Because problem solvers often must make decisions in the absence of complete information, errors are inevitable. This is especially true of problem solvers whose performance is subject to the uncertainties of external evaluation. These problem solvers require feedback to evaluate their decision-making performance. Otherwise they can neither learn from their experience nor attempt recovery from failure, since they will be unaware of their failure to reach a satisfactory solution.

Failure recovery is the process of recognizing that events fail to meet expectations, explaining the cause of the failure, and taking the appropriate actions to remedy the knowledge that led to the failure. The failure recovery process does not include the finding of another solution, that is accomplished after a problem solver's reasoning has been repaired. We treat recovery from failure as another instance of the problem solving process previously specified. In this instance of our problem solving model, the new problem is the failure of the problem solver's reasoning to successfully resolve the original problem. This "new" problem must be understood (i.e., the faulty knowledge or reasoning rule must be found) and a plan for its solution determined (i.e., the faulty knowledge must be corrected). All the components of problem solving previously discussed, including case-based reasoning, can be employed to repair the problem solver's knowledge and permit the original problem solving process to continue to a successful conclusion. Case-based reasoning applies to failure recovery in a way that is analogous to the way it was used in the original problem domain:

1. previous similar failures can suggest which reasoning step was used incorrectly (i.e., the failure is understood)

2. previous similar failures can suggest ways of correcting the knowledge leading to the failure (i.e., a plan is generated for correcting the erroneous knowledge)

Recovery from failure plays two roles in our model of adaptive problem solving. First, it directs the repair of a problem solver's knowledge so that the current problem solving effort can be completed successfully. It also acts as the learning element in a self-improvement process. In this second role, "lessons" learned from a failure are recorded so that they can improve the problem solver's performance on future problems. Our primary focus in this chapter is on the first role of failure recovery, since this role provides another opportunity to demonstrate case-based reasoning. We address the second role briefly at the end of the chapter.

In the sections that follow, we will first provide a brief overview of failure recovery. This overview will identify processes analogous to those discussed in the previous two chapters, as well as processing components unique to this problem solving context. We will then discuss in detail each component of failure recovery in a separate section. This discussion will point out the different knowledge and reasoning used in each process. As we will show, even though this knowledge is different (i.e., it is knowledge about the problem solver's reasoning), it is still compatible with our general model of problem solving and thus can be improved by case-based reasoning.

5.2 Overview of failure recovery

Failure recovery is that instance of problem solving concerned with correcting some detected failure (problem) in the problem solver's reasoning. When viewed this way, failure recovery is a form of "meta-problem solving" (Stefik, 1981), i.e., reasoning about the problem solving process. The problem solver's goal during failure recovery is to discover the error that explains his failure to solve the original problem and to attempt to remedy the the knowledge leading to that error so that problem solving can continue. Failure recovery begins when a test of the predictions from planning fail to match the actual results provided by feedback from the problem environment. With an error recognized, the problem solver must attempt to understand the error and generate a remedy for it. This done, the problem solver can test the remedy by returning to the original problem context to once again attempt its solution. In figure 5-1 below, this portion of the problem solving process is shown in boldface.

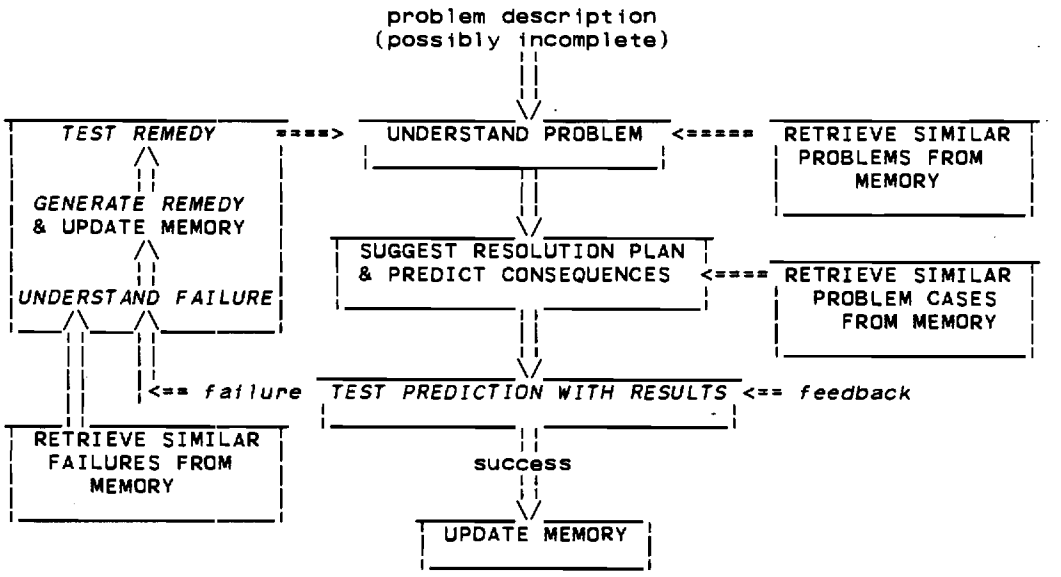


Figure 5-1

As indicated above, there are four stages in our model of failure recovery. First, a problem solver must recognize that failure has occurred in the problem solving process. This recognition causes another instance of problem solving to be created which includes new understanding, generation, and test phases oriented specifically to the repair of the problem solver's reasoning. In general, we call this new instance of problem solving *"remediation"* to distinguish it from the original problem solving effort which failed and to emphasize its role in repairing the problem solver's reasoning. As an instance of our problem solving model, remediation includes analogous components for understanding failures (i.e., assigning blame) and planning for their resolution (i.e., selecting the appropriate remedy).

In the first stage of failure recovery, feedback from the problem environment is evaluated to determine that indeed there has been a failure of the problem solving process. This decision results primarily from testing the predicted versus actual results of plan execution. In some cases, a problem solver's failure may be pointed out explicitly by agents who externally evaluate the results and provide feedback. As we will see later, even when failure is explicitly indicated, we must have some internal evaluation process to guard against errors introduced by the evaluation process itself. The window dispute provides an example of a problem solver being told explicitly of her failure to resolve the problem. In other cases, such as Orange-Dispute-0, the problem solver must internally evaluate feedback to determine that a failure has occurred.

Once a problem solver decides that a failure has occurred, either by internal or external evaluation, the process of remediation begins. The first stage of remediation, as an instance of problem solving, is understanding. In remediation, a problem solver's interest is first focussed on understanding the cause of the failure. This stage of remediation is sometimes referred to as "blame or credit assignment" (Minsky, 1963) or "failure explanation" (Schank, 1982; Sussman, 1975). As a result of this stage of remediation, a problem solver should be able to identify a specific part of his reasoning as the likely source of error. If we decompose a problem solver's reasoning into stages (as we have in the preceding chapters), and if broad classes of errors are known, then "blame assignment" is equivalent to classifying the failure into known classes of decision error, where each of these classes corresponds to a stage of the problem solving process. For example, the librarian in the window dispute probably realized that her assumptions about the men's goals were a likely source of error. In our framework, this would indicate that she classified the failure as resulting from an error in understanding (i.e., a "misunderstanding" type failure). In particular, she decided that a reasonable explanation for her failure to resolve the dispute was that she had incorrectly inferred the men's goals. As we discussed in chapter three, this classification decision is equivalent to "hypothesis formation" in diagnostic problem solvers.

Once a problem solver has classified the failure, specific remedies are available to deal with known types of failures during the second stage of remediation. This is the third stage of failure recovery, but the second stage, the planning stage, of remediation. Remedies play an analogous role in the repair of a problem solver's knowledge as plans do in solving domain problem. The difference, of course, is that remedies operate on the problem solver's knowledge and reasoning, not on the original domain problem. In the window dispute during this stage, the librarian selected and applied a remedy that we call "ask for goals directly." This remedy suggests that when you suspect that you do not understand someone's goals, ask them directly to tell you their goals. That is what the librarian did in the window dispute. Within our framework, this remedy allows a problem solver to alter its internal representation of the problem. In this particular case, we assume that, having learned the men's real goals, the librarian incorporates the new information into her internal representation of the men's goals. In general, the generation and execution of a remedy results in a specific change to the problem solver's previous knowledge. We will present several remedies in a later section to illustrate this point.

The last phase of failure remediation is an optional test of the remedy. This stage is useful when the problem solver has several equally plausible remedies. Rather than guessing, a problem solver can obtain additional information (test the external environment) to help in selection of the best remedy. For example, in her analysis of the window dispute, the librarian could have decided to guess the men's alternate goals. When she settled on a reasonable goal, she could ask the men to comment on her guess (i.e., "Do you want the window open because ...?"). Such questions allow the problem solver to "test" a reactive environment in a specific way that aids a problem solver's reasoning. This testing is not essential, since the problem solver tests the remedy anyway by returning to the context of the original problem when remediation is complete. If the original problem is successfully resolved, then the remedy was correct. The original problem always provides the ultimate test of remediation. The testing stage of remediation is thus considered an optional process which can minimize the number of failure recovery passes.

The four stages of failure recovery discussed above are represented graphically below;

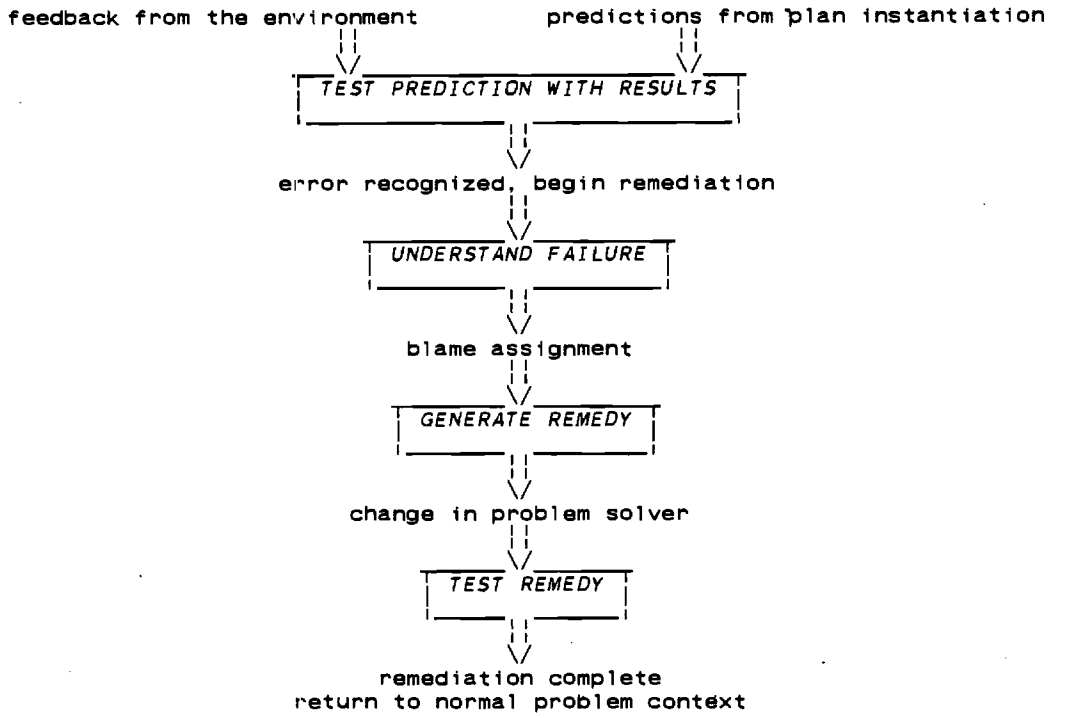


Figure 5-2

With this description of failure recovery, we are now ready to overview how case-based reasoning supports this process. First, it is important to note that, for remediation, the "new" problem includes the various features of the original problem plus any failed solution attempts. In other words, the new problem is the entire case history built during previous problem solving efforts. In the window dispute, for example, the librarian now knows that the dispute cannot be resolved by the "split the difference" plan. These additional features, the plan attempted and feedback from the environment, become cues that serve as memory retrieval probes to facilitate reminding of a previous similar failure. A recalled failure allows case-based reasoning to be used for failure recovery in ways analogous to the normal problem solving context. Thus during failure understanding, a similar failure can suggest a plausible cause for the failure of the current case. During remedy generation, a similar failure can suggest a plausible remedy which can be used to repair the knowledge that led to the current failure.

To illustrate case-based reasoning in failure recovery, let us return to the window dispute case. When the librarian realized she had failed to resolve the dispute, she was reminded of the day before when she had also failed to resolve a quarrel between her daughters (i.e., Orange-Dispute-0). In that case, she had also attempted to apply a version of the "divide equally" plan and it was also unsuccessful. Guided by that recalled case, she wonders if perhaps she might also be misunderstanding the real goals of these men quarreling over the window (i.e., her attention is drawn to a plausible explanation for the failure which provides an understanding). After all, she had misunderstood one of her daughter's intentions with the orange. She would have been smart to ask her daughters their intentions before offering her advice (i.e., she considers the plan she should have used in the recalled failure to remedy her knowledge). Using this reasoning, the librarian decides to ask the men about their goals. The role of case-based reasoning in failure recovery is summarized in Figure 5-3 below.

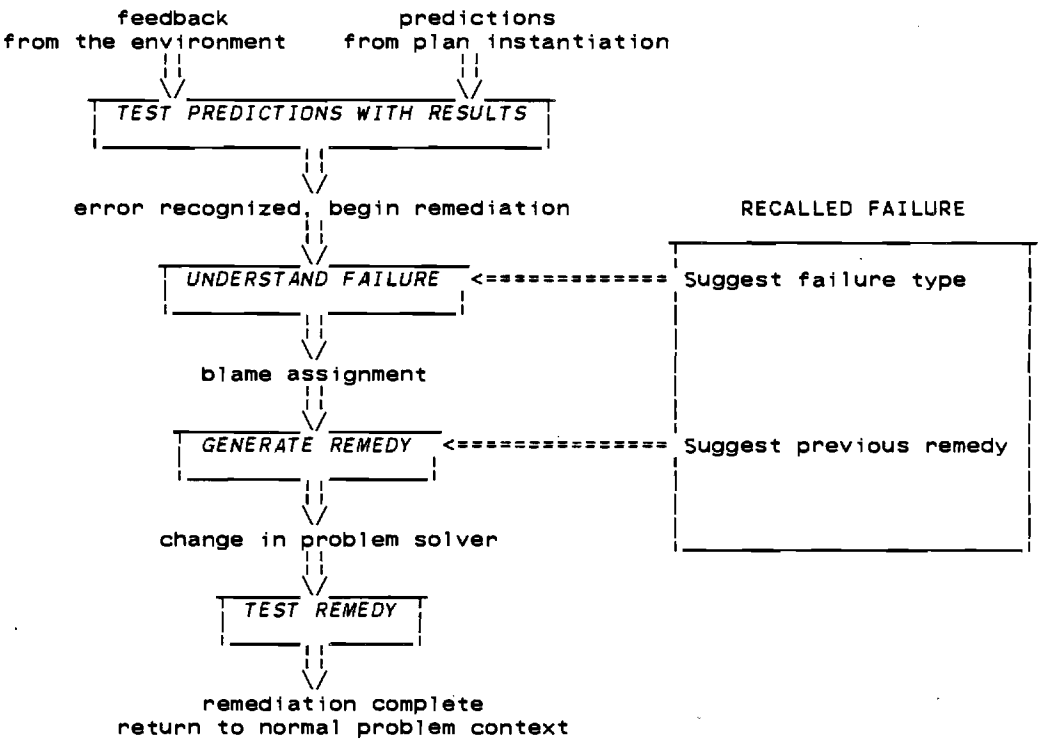


Figure 5-3

5.3 Evaluating performance in the advisory role

Consider for a moment what our automated problem solver is up against in trying to evaluate its own performance. It only knows what it is told about a problem and usually has no way of directly inspecting or otherwise changing the given problem. On top of that, the problem does not belong to the problem solver, it really belongs to the problem solver's clients. We will use the term client to designate the end user of a problem solving system. The clients are the final judge of the problem solver's performance. This, in general, is common to all problem solvers in an "advisory role" (Gershman, et al., 1984; Haefner, 1984).

The advisory role differs from many problem solving situations because the environment, in the person of various clients, is not constrained to provide consistent information that can be used to learn an absolute performance standard (Ward and McCalla, 1982). Compare giving advice on resolving disputes to playing checkers or chess, for example. These games have deterministic rules that can be used to build a separate analytical performance standard that can be used to automatically evaluate the decisions of a problem solver.* Even in common sense dispute resolution, there is always an element of uncertainty during performance evaluation because the clients and their disputes are rarely exactly the same. Since the clients are the ultimate judges of performance for each case, a problem solver's general performance standard is subject to individual differences among clients.

---

*For example, Samuel (1963) used one version of his checkers program as a performance standard, while playing another version of the program that was learning. If the learner won, then it became the new performance standard.

Our approach to this element of uncertainty in performance evaluation is twofold. First, we provide our problem solver with a method of default reasoning that tests whether results match predictions.* Components of this evaluation process will be presented in a later subsection. Our second approach to the uncertainty of the advisory role is to explicitly seek feedback from the clients at critical evaluation points. We thus acknowledge the unavoidable fact that a problem solver in the advisory role will never know about his performance until told by the clients or provided some other external notification.

Recognizing these two complementary approaches allows us to identify two different types of feedback that support a problem solver's performance evaluation process:

1. Feedback signalling success or failure from external evaluation by the clients

2. Feedback of results or advice from the clients that must be subjected to more extensive processing internally.

The first type of feedback is essentially a success or failure message from the clients. With a success signal, there are several possible courses of action. A problem solver could stop at this point, credit this case to the success column, and figure there was nothing to learn from this case. Such an approach is short sighted, in our opinion. We contend that a problem solver must also seek feedback of the second type that relates specific results for additional internal evaluation. If the internal evaluation indicates a failure, then we have detected the possible occurrence of a "false positive." If the results confirm expectations, then a problem solver has some assurance that the "success" was indeed a successful case of problem solving.

When the signal from the clients indicates a failure, a problem solver also seeks feedback of the second type; this time in order to provide guidance to remediation.** Thus, failure recovery can be initiated from both explicit feedback from the environment (i.e., the clients say so), as well as from a recognized failure during the internal evaluation of feedback. The next two sections discuss these two interrelated aspects of using feedback in the evaluation process.

---

*This provides a simple performance evaluation capability that clearly depends on the level of match and inference sophistication (Charniak, et al, 1980). When the problem solving situation permits, an objective prediction provides half the essential elements of this performance standard. For example, in Samuel's checkers program the standard version provided the prediction. It is the match of the performance standard with the feedback results that is often described as "self-awareness." Providing a problem solver with the "self-awareness" to recognize that a failure has occurred is an essential component of a self-adaptive problem solver. This capability is necessary to protect a problem solver from being swayed by "false positive" cases (i.e., when a problem solver succeeds but for the wrong reason).

**We do not address the recognition of "false negatives," the complementary issue to recognition of "false positives." This decision was influenced by our observation that when a problem solver, in the advisory role, suspects a "false negative" (i.e., the reported failure is really a success), then a problem solver usually needs to engage in dialogue to "persuade" (Sycara-Cryanski, 1985) the clients that they are wrong. The subsequent dialogue and argumentation can be viewed as another instance of problem solving which may offer other uses of case-based reasoning. This is an area for future research.

5.3.1 Requesting feedback from external evaluation

The first use of feedback occurs when a problem solver explicitly requests an evaluation of its performance from its clients. A problem solver, in the advisory role, needs this feedback to answer the most basic performance evaluation question of all: "Did I succeed or fail?" The response from the clients can take many forms, but the essential message is an indication of success or failure. An obvious example of this behavior occurs in the MEDIATOR when a mediation plan has been selected and instantiated. The client is asked to determine the plan's acceptability and provide success or failure feedback, as illustrated in Figure 5-4 below:

I/O BEHAVIOR OF THE MEDIATOR REQUESTING FEEDBACK FROM EXTERNAL EVALUATION

```
      :
      :
I suggest that the plan called "one cuts the other chooses" be used.
Do you agree, that this is the best solution? (Y or N) No.
   **** "one cuts the other chooses" not acceptable ****

Can you provide any comments that might help me remedy this failure?
(i.e., What happened?)
      :
      :
```

Figure 5-4

In this example, the initial request for evaluation from the client resulted in feedback that indicates that the mediator's resolution attempt was a failure. This keys an explicit failure recognition and initiates failure recovery as a result of external evaluation.

At the risk of oversimplification, we will consider only two alternative responses to a problem solver's request for external evaluation. As illustrated in Figure 5-4, the client either signals success or failure (i.e., yes or no). When a problem solver receives a failure signal (NO), its initial focus should be on obtaining advice or other information (such as the client's objections) concerning the nature of the failure. In later sections, we will see that the problem solver uses the information provided by the clients to guide remediation.*

When a problem solver requests external evaluation, the other possible response by the client is an indication of success (YES). This is illustrated by the following fragment from the MEDIATOR program shown in Figure 5-5.

I/O BEHAVIOR OF THE MEDIATOR RECEIVING POSITIVE FEEDBACK FROM EVALUATION

```
      :
      :
I suggest that the plan called "one cuts the other chooses" be used.
Do you agree, that this is the best solution? (Y or N) Yes.
Do you know the results of the plan's execution? (Y or N) Yes.
Please indicate the results:
      :
      :
```

Figure 5-5

---

*We have considered using client feedback only within the context of our overall failure recovery mechanism. The next step in this process is understanding the feedback as part of the original problem and resolution failure. We do not address this in detail, but to the extent that we have, we see client feedback functioning in the limited role of providing remedy guidance to a problem solver. Thus, in the remedy generation stage discussed later, advice provides explicit information that directs the repair of the MEDIATOR's reasoning. Learning from advice has been described (Cohen and Feigenbaum, 1982) as involving the following five processes: (1) request -- request advice from the expert, (2) interpret -- assimilate into internal representation, (3) operationalize -- convert into usable form, (4) integrate -- integrate into knowledge base, (5) evaluate -- evaluate the resulting actions of the performance element. In our formulation, the interpretation and operationalization processes are part of the understanding process, while integration is roughly equivalent to our term for remedy generation and the subsequent storage of this case into our conceptual memory of case experiences.

Notice that we have used explicit success or failure signals in our algorithms. As figure 5-5 shows, a problem solver should be interested not only in learning that the suggestion provided acceptable advice, but also that the advice produced the expected results. Toward this second end, we see the MEDIATOR above ask for explicit feedback on the results. When this feedback is received, it allows a problem solver to perform its own internal evaluation. This internal evaluation is essentially a match of the results with prior predictions. For feedback to be useful to a problem solver, it must be recognizable as a type of "success" or a type of "failure." As a result of this evaluation of feedback, a problem solver may initiate failure recovery even when the client has indicated success.

5.3.2 Matching predictions with results

In order to recognize that feedback represents success or failure, a problem solver needs something against which the feedback can be compared. The feedback provides only half the information from which a problem solver must internally derive success or failure classifications for the current problem solving effort. The other half comes from the predictions developed during the planning stage of problem solving. In its most elementary form, this process may be represented as a simple match between expected and actual events (Charniak, et al., 1980). However, this recognition process is far from simple. For example, a partial match can be a success or a "near miss" failure (e.g., Carbonell, 1979; Winston, 1975).

To illustrate this problem, let us assume, for the moment, that the problem solver does not generate predictions of the plan's expected effect on the client's goals and instead uses the client's goals as one element in comparison with the reported results. In Candy-Dispute-0, for example, if we infer that the boys each have the goal of eating the whole candy bar, then the "one cuts, the other chooses" plan cannot be recognized as a success, since neither one of the boys achieved his goal. If we simply match each child's desired goal state to the reported results of the "one cuts the other chooses" plan, as shown in Figure 5-6, we would not evaluate it as achieving the expected partial goal satisfaction. It is for this reason that we must record the effects of plan execution on the disputants' original goals as a prediction.

SIMPLE MATCHING OF DISPUTANT GOALS WITH RESULTS DOES NOT WORK

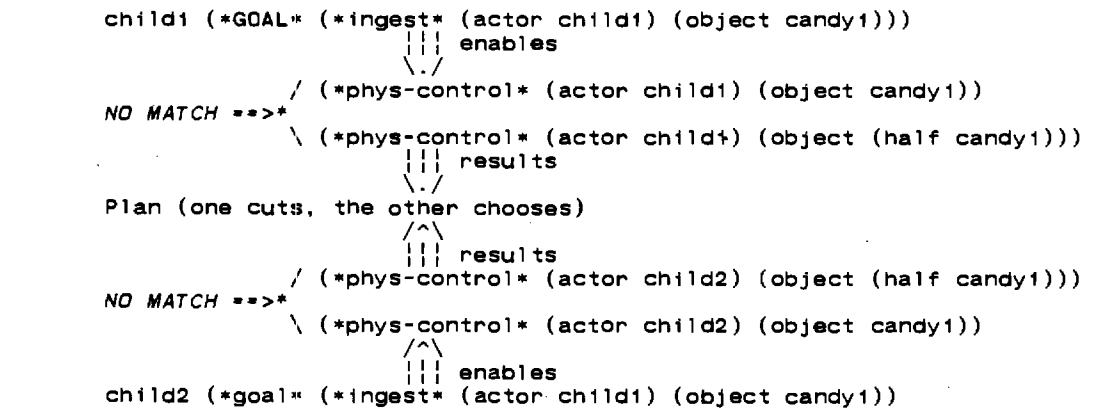


Figure 5-6

In a very real sense, the capabilities of an adaptive problem solver hinge on how well this problem of partial matching is resolved. If the match algorithm is too optimistic (i.e., accepts any close matches), then failures will not be detected and a problem solver will learn "bad lessons." If the match algorithm is too pessimistic (i.e., accepts only exact matches), then much work will be wasted trying to solve already solved problems and learning opportunities will be lost.*

---

*We recognize the importance of this matching and also realize that, at least in people, matching (i.e., recognition) is context dependent. For example, see Green and Swets (1966).

In an attempt to bound this issue, we have identified two complementary, but opposite approaches. One approach accounts for goal satisfaction or failure via a pessimistic algorithm that essentially says "If the results are not recognizable as some known form of success, then it must be a failure." Carbonell (1979) has described how this type of heuristic can be employed in the classification of events as either success, partial success, or failure. Our adaptation of Carbonell's algorithm is presented in Figure 5-7:

A PESSIMISTIC MATCHING ALGORITHM FOR RECOGNIZING SUCCESS OR FAILURE

IF the *initial* situation is judged to be less than the *resultant* situation as reported from feedback which is equivalent to the *desired* situation
THEN the event is recognized as a *success*.
ELSE

IF the *initial* situation is judged to be less than the *resultant* situation as reported from feedback which is judged to be less than the *desired* situation
THEN the event is recognized as a *partial success*.
ELSE the event is recognized as a *failure*.

Figure 5-7

Applying this algorithm to Candy-Dispute-0, we note that initially neither boy has any candy and both want it, so from both their perspectives their goals have an initial quantitative value of 0. Similarily their goals each have a desired quantitative value of 1. From the algorithm, it is easy to see that no assignment of the whole candy bar to either child will result in both children's evaluation of the action as a success. On the other hand, since half a candy bar (or any portion for that matter) is quantitatively greater than either child had initially, the results of compromise plans like "one cuts, the other chooses" can be evaluated as a success even when they do not literally achieve the desires of a disputant.

A second approach to the problem of recognizing partial matches of execution results is based on a somewhat more optimistic view. It has been adapted from the work of Flowers (1982) on recognizing contradictions in argumentation.* Basically, the idea is to consider any result that does not contradict the plan's predictions to be a success. Instead of trying to confirm the result as a type of success, this algorithm focusses on trying to recognize failures. It has been employed previously as part of the process for inferring goal relationships, as discussed in section 2.3.3. This algorithm is shown in Figure 5-8:

AN OPTIMISTIC MATCHING ALGORITHM FOR RECOGNIZING SUCCESS OR FAILURE

IF the results from feedback

(1) *negate* the plan's predictions or
(2) *contradict* the plan's predictions or
(3) lead to an *inferred contradiction*

THEN the event is recognized as a *failure*.
ELSE the event is recognized as a *success*.

Figure 5-8

---

*The technical meaning of the term "contradiction" is different here than in chapter three. Since we provide examples on the next page, we hope to avoid possible confusion over different implementations for the same corresponding concept. In chapter three, we were concerned about the mutual consistency of separate inferences by a problem solver. In that context, contradiction stood for singly consistent but mutually inconsistent inferences. Here we are using the term to apply to the concept of confirming that events are not inconsistent with an interpretation of success. These two processes are close enough to warrant the same English name, but occur at different points and for different reasons in our model.

Using the approach indicated in Figure 5-8, we note that use of the "one cuts the other chooses" plan for the candy dispute produces the prediction that each boy will eat his half of the candy. When subsequent events are reported as feedback, they are matched against those predictions. Failure can be matched one of three ways corresponding to each of the concepts "negation," "contradiction," and "inferred contradiction." The term "negation" means the results do not logically negate either of the disputant's predicted actions. "Boy1 did not eat his half of candy1" reflects a direct negation of boy1's predicted action. The term "contradiction" means that the results fail to match the predictions with respect to their components: the expected action, its roles and fillers (this is defined as micro-match by Charniak, et al. (1980) and Schank and Riesbeck (1981)). "Boy1 ate apple1" contradicts the predictions concerning the object of boy1's action. Finally, the term "inferred contradiction" means the results imply a contradiction to a precondition for the plan's prediction. "Boy1 has half of candy1" is inferentially consistent with the plan's predictions even though it does not match, because it is a precondition of boy1's predicted ingest action. Using this approach, a subsequent event is given the benefit of the doubt in confirming a problem solver's predictions "unless it contradicted" the expected results according to one of these categories of failure.

As mentioned earlier, both the optimistic and pessimistic approaches to success and failure recognition have serious problems when used in isolation. Each approach seems to adopt one extreme view of a problem solver's subjective acceptance criterion. In its most extreme case "pessimistic evaluation" would computationally represent a problem solver who minimizes "false alarms" and accepts only solutions that are known to satisfy the stated expectations explicitly. On the other hand, "optimistic evaluation" represents a problem solver who minimizes "correct rejections" and discards only those solutions known to be failures.

We recognize the importance of this issue, but have no general solution to offer. In the MEDIATOR, the match is explicitly controlled using an optimistic approach. This was done because we normally wanted to emphasize or investigate some other part of the reasoning process. As part of the design of each case, we would construct the M-POSSIBLE-EVENTS component of the case so that the match would correspond to the failure behavior of interest. This was an acceptable expedient during the development and testing of other parts of the program, but further work is required if case-based problem solvers are to be fully automated.

5.3.3 MEDIATOR's performance evaluation algorithm

The MEDIATOR combines these ideas into an overall performance evaluation algorithm that seeks answers to the following three questions:

1. is the suggested plan acceptable?
2. if acceptable, do the results match expectations?
3. if unacceptable, why was the plan unacceptable?

In question number two, the match that is performed can be either optimistic, pessimistic, or some combination of both as discussed in the previous subsections. The algorithm shown in Figure 5-9 describes the evaluation process used in the MEDIATOR program and indicates the complementary uses of internal and external evaluation independent of the matching problem:

THE MEDIATOR'S OVERALL EVALUATION ALGORITHM

```
ASK disputants to externally evaluate acceptability of plan
IF the proposed plan is acceptable
THEN ASK if disputants know the results of plan execution

     IF the results are known
     THEN ASK for the results and evaluate internally
        by matching the results with those predicted
        from contract instantiated by selected plan
        IF match is successful
        THEN reinstantiate the mediation case as a confirmed
           success and store the case in memory
        ELSE (match has failed indicates possible false positive)
           reinstantiate the mediation case as unsuccessful and
           store the case in memory after attempting failure recovery
     ELSE (results are not known) reinstantiate the mediation case
       as an unconfirmed success and store in memory

ELSE (plan not acceptable) ASK for feedback that can be used to
  direct failure recovery, reinstantiate the mediation case as
  unsuccessful and store in memory after attempting failure recovery
```

Figure 5-9

The MEDIATOR recognizes failures as a result of external or internal evaluation. In either case the program notes its failure by reinstantiating the mediation case as an "unsuccessful mediation." We will illustrate this in the following sample output from the program during the Sinai dispute case. If you recall from chapter one, a problem solver has read in the paper about Israel and Egypt fighting over the Sinai. This causes it to be reminded of the Korean War which also involved fighting over land. Based on this reminding, the problem solver predicts that Israel and Egypt will divide the Sinai equally, (what was done in Korea). This advice is rejected by the disputants during external evaluation, with feedback provided as indicated in Figure 5-10 below:

I/O BEHAVIOR OF MEDIATOR RECEIVING FEEDBACK FOR FAILURE RECOVERY

```
(mediator sinai-dispute t)
Considering the following problem:
   Israel and Egypt both want the Sinai,
   which has been presented as ako M-DISPUTE.
      :
      :
I suggest that the plan called "one cuts the other chooses" be used.
Do you agree, that this is the best solution? (Y or N) No.
   **** "one cuts the other chooses" not acceptable ****

Can you provide any comments that might help me remedy this failure?

((*MTRANS* (ACTOR ISRAEL)
           (MOBJECT (*GOAL* (*NAT-SECURITY* (ACTOR ISRAEL)
                                            (OBJECT SINAI)))))
[i.e., Israel says they want the Sinai for national security.]
 (*MTRANS* (ACTOR EGYPT)
           (MOBJECT (*GOAL* (*NAT-INTEGRITY* (ACTOR EGYPT)
                                             (OBJECT SINAI))))))
[i.e., Egypt says they want the Sinai for national integrity.]

Attempting to explain this failure and find a new solution.

Considering the following problem:
   failed mediation for Israel and Egypt both want the Sinai,
   which has been presented as ako M-UNSUCCESSFUL-MEDIATION.
   It will be referred to as #<M-UNSUCCESSFUL-MEDIATION 40544074>
      :
      :
```

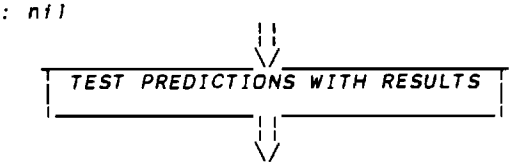
Figure 5-10

As shown in Figure 5-10, feedback is provided in a conceptual dependency style to simplify the interface for the program and minimize the difficulties in "operationalizing" (Mostow, 1983) the advice. The ability of a problem solver to use feedback effectively is dependent on whether the feedback is at an abstract or concrete level. Abstract feedback must be made "operational" and concrete feedback requires knowing where and how to make the appropriate use of the information. These considerations, as well as other issues of natural language processing of input to the problem solver, have required a simplified interface to provide feedback. So for practical reasons, the feedback provided the program is already in a representation acceptable to the program.

At the bottom of Figure 5-10, the MEDIATOR is beginning failure recovery. Notice that the program's behavior at the beginning of failure recovery reflects the recursive use of the problem solving model. Some indication of this can be seen by comparison to its behavior at the top of Figure 5-10. Notice too that the problem for failure recovery is now identified as an M-UNSUCCESSFUL-MEDIATION instead of a M-DISPUTE as it was earlier. As indicated in the algorithm described in Figure 5-9, the mediation case is reinstantiated after a failure has been recognized. This transformation of a case including the addition of feedback is illustrated in Figure 5-11:

CASE REINSTANTIATION AS A RESULT OF FAILURE RECOGNITION AND FEEDBACK

```
M-MEDIATION isa M-PROBLEM
    dispute: sinai-dispute
    mediation-plan: M-ONE-CUTS-OTHER-CHOOSES
                    expected-contract: M-DIVIDED-OBJ-CONTRACT
    predictions: M-POSSIBLE-EVENTS
                 results-ok:
                    ((*phys-control* (actor ISREAL)
                                     (object (half Sinai))
                     (*phys-control* (actor EGYPT)
                                     (object (half Sinai))
    results-confirmed: nil
    feedback: nil
```




```
M-UNSUCCESSFUL-MEDIATION isa M-MEDIATION isa M-PROBLEM
    dispute: sinai-dispute
    mediation-plan: M-ONE-CUTS-OTHER-CHOOSES
                    expected-contract: M-DIVIDED-OBJ-CONTRACT
    predictions: M-POSSIBLE-EVENTS
                 results-ok:
                    ((*phys-control* (actor ISREAL)
                                     (object (half Sinai))
                     (*phys-control* (actor EGYPT)
                                     (object (half Sinai))
    results-confirmed: nil
    feedback: ((*MTRANS* (ACTOR ISRAEL)
                         (MOBJECT (*GOAL* (*NAT-SECURITY* (ACTOR ISRAEL)
                                                          (OBJECT SINAI)))))
               (*MTRANS* (ACTOR EGYPT)
                         (MOBJECT (*GOAL* (*NAT-INTEGRITY* (ACTOR EGYPT)
                                                           (OBJECT SINAI))))))
```

Figure 5-ii

It is this final representation of the mediation case as an "unsuccessful mediation" that is used by failure recovery as its "new" problem.

5.4 Understanding failures

Once a failure has been recognized, the problem solver needs to understand the failure. In our model, this means it must classify the failure into one of a set of known failure types. Failure classification is essentially the blame or credit assignment task (Minsky, 1963). Because failure understanding in our model is analogous to problem understanding, it also involves failure elaboration as well as classification. We will concentrate only on failure classification in the following discussion.

We have identified five general classes of failure:

1. misunderstanding failures
2. planning failures
3. plan execution failures
4. evaluation failures
5. unsolvable problem failures

Each of these failure types is named so as to describe the type of error that can lead to failure. Misunderstanding a problem might result in the selection of an ineffective plan. A plan may fail because of a planning error such as choosing the wrong planning policy. A good plan may fail because of bad execution or because a random event caused an otherwise good plan to fail. There may be no way to resolve the problem (a no win situation).

Any one of these failure classes can be arbitrarily hard to determine. Of the five general types noted above, we will concentrate on only those of the first two types. The failures of the third type, which involve execution monitoring, have been the primary focus of earlier failure recovery research (Sacerdoti, 1977; Ward and McCalla, 1982; Wilkins, 1984). As discussed in the previous section, we do address the recognition of "false positives," which are one type of evaluation failure (i.e., type four above). Failures of the fifth type are interesting because, in general, we want problem solvers to know when they cannot solve a problem and not waste effort trying. This is a much desired capability in a problem solver,

but one we do not address.

Failures due to misunderstanding can occur either when a problem solver infers the wrong problem type or infers portions of the representation incorrectly. Notice that both of these failure types correspond to stages of our model of problem understanding. For example, if a mediator inferred that a dispute was a "physical dispute" when it was in fact an "economic dispute," then we would have a misunderstanding failure due to a "misclassification." The other type of misunderstanding failure is the result of incorrect elaboration by a problem solver. In Orange-Dispute-0, for example, the mother-mediator incorrectly inferred her daughter's goal. This is an example of a miselaboration due to "wrong goal inference."

Failures due to bad planning can occur either when a problem solver chooses the wrong planning policy, when an inappropriate plan is selected, or when the plan is instantiated incorrectly. Here too, the classes of failure correspond to each of the stages of our model of planning. For example, if a mediator suggests a compromise plan and the disputants want an "all or nothing" solution, then the failure was due to a "wrong planning policy." Bad planning is usually associated with the selection of an incorrect plan. This is more likely to happen when a problem solver is learning plan applicability conditions. With experience, a problem solver gradually acquires the knowledge to avoid failures due to "wrong plan selection" or "wrong plan instantiation."

A portion of the MEDIATOR's generalization hierarchy for failures is shown in Figure 5-12.

A PORTION OF THE FAILURE ABSTRACTION HIERARCHY

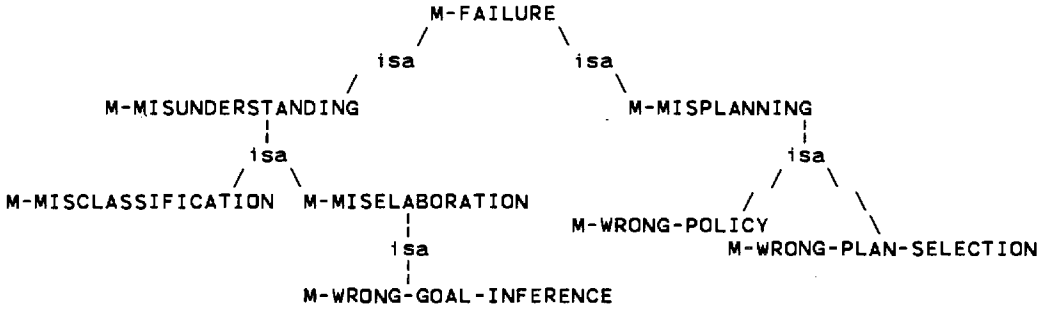


Figure 5-12

Case-based reasoning supports failure understanding by the recall of previous cases which share similar failure features. For example, the recalled case may have involved a similar dispute or experienced a failure of the same plan. If the failure classification is appropriate for this failure according to known constraints, then that failure classification can be transferred. This provides a heuristic explanation (blame assignment) for the failure based on analogy to a previous case. In the following sections, we first discuss the reasoning required to track down and explain errors, and then show how previous cases can help.

5.4.1 Failures due to misunderstanding

Failures that are the result of a misunderstanding (misrepresentation) of the problem are a very interesting and likely source of individual differences in problem solving behavior (Hayes, 1981). As discussed in chapter three, there are many inferences made by a problem solver during the initial construction of a problem representation. Any one of these inferences have the potential to cause a failure. In order to help recall these inferences, the overall view of the understanding process in our model of problem solving is shown once again in Figure 5-13:

REVIEW OF THE OVERALL PROBLEM UNDERSTANDING PROCESS

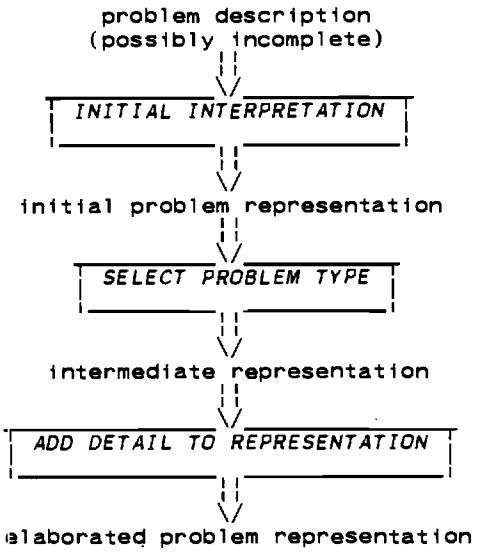


Figure 5-13

Our specific ideas on default failure detection borrow heavily from the notions of truth maintenance and non-monotonic reasoning (Doyle, 1979; O'Rorke, 1983). This is accomplished by tracing causal inference dependencies and reconsidering past problem solving decisions in light of feedback (i.e., new information). In many AI problem solving systems (e.g., de Kleer, et al., 1979; Doyle, 1979; Stefik, 1981; Sacerdoti, 1977), a detected failure in reasoning, usually the result of some constraint violation, requires the problem solver to retrace its reasoning and retract any contradictions. In the worst case, this might require restarting at the very beginning. But, for all those cases where the wrong assumption makes very few changes in previous problem solving steps, the preferred approach is to reaccomplish only the minimum steps necessary. In procedurally-oriented problem solvers, two techniques have been developed to address this problem: "queue-based control" and "dependency-directed backtracking" (de Kleer, et al., 1979; Doyle, 1979; Stallman and Sussman, 1979). These techniques depend on the fact that (1) not all inferences are of equal weight thus the queues provide a means of directing recovery processes based on an a priori ranking of inferences, and (2) some organizational conventions are followed to record the "chain" of inference dependencies. Recognizing these facts has led us to adopt an equivalent yet different approach in our research.

As part of the explicit organization of inferences, we have previously identified different actions that collectively make up the understanding task. These individual subtasks provide a finer grain of detail within which to classify failures. For example, all inferences made during the problem classification stage of problem understanding are collectively known as "contextual inferences" and form one category of potential error. This type of error might occur in the MEDIATOR, when a retrieved case suggests a dispute classification (e.g., "physical dispute") that is wrong (it was really an "economic dispute"). This initial misunderstanding will cause the problem solver to bias later reasoning such that an incorrect plan will be suggested (imagine telling an orange vendor and his customer, who are quarrelling over the price of an orange, that they should resolve their dispute by "one cuts the other chooses").

Elaboration inferences provide another source of potential errors during understanding. For example, the most important type of elaboration inference is goal inference. Inferring goals is a classic source of misunderstanding (Fisher, et al., 1981). For instance, people presented with the ill-defined description of Orange-Dispute-0 (i.e., "Two sisters are quarreling over an orange.") invariably make the "wrong goal inference" that the sisters each want to eat the orange. This misunderstanding leads them, quite naturally, to suggest "divide equally" type solutions just as the MEDIATOR does.

Our model also makes explicit the dependence relationship between inferences. For example, we specified in chapter three that classification decisions would be made before problem elaboration. This means that, within the understanding task, elaboration inferences depend on the context provided by the problem classification decision. Noting the relationship between contextual inferences and elaboration inferences permits failure recovery actions to be directed so that only the appropriate portions of the internal case representation need be examined and corrected by the appropriate remedy. This effectively produces a focus for failure recovery comparable to that admired in queue-based control

techniques. In addition, this inference relationship in conjunction with our problem solving model allows a natural variation of dependancy directed backtracking. If a goal inference is identified as being in error, then the surrounding contextual inference need not be reconsidered. But if a contextual inference is in error, then all elaborational inferences will need to be reconsidered since they are dependent on the classification decision.

In situations where there is neither external information to guide failure understanding nor remembered cases of similar failure from which to reason, the dependence relationship of these inference types permits default investigations based on "backtracking" first with respect to elaborational inferences then to any classification inferences. When the MEDIATOR program is faced with an understanding failure without feedback, it directs its information seeking activity toward verifying its goal inferences. If that fails to yield any changes, the next area of inquiry is the examination of classification decisions.* Figure 5-14 illustrates the sequence of investigation that we use in directing the analysis of failures to determine if they can be attributed to understanding failures.

BACKTRACKING SEQUENCE IN DEFAULT CLASSIFICATION OF UNDERSTANDING FAILURES

Figure 5-14

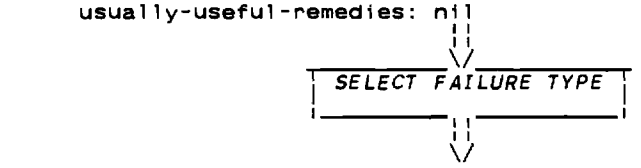
Because we view failure classification as exactly the same process as problem classification, we need to illustrate exactly how the failure representation is altered by its classification. To illustrate the effect of the failure classification process, we show in Figure 5-15 the classification of Orange-Dispute-0 failure as one due to "wrong goal inference." As you may recall from this case, the mother decided she had misunderstood her daughter's goal after observing (receiving feedback) the results of plan execution. Notice that besides assigning blame to goal elaboration by instantiating the unsuccessful mediation case as a M-WRONG-GOAL-INFERENCE, the "usually useful remedies" for such failures are also indicated at the bottom of the figure. This is analogous to the "usually-useful-plans" slot that provided the corresponding information for dispute classification (see section 3.2.2).

---

*Donald Norman relates an observation about people who walk up to their cars and discover that their keys don't work. He reports that they always seem to first assume that they have the wrong key or their car lock is malfunctioning. Only after checking these explanations, do they begin to suspect that the car might not be their car. This behavior corresponds to our notion that lower level classes of failure, i.e. elaboration errors, are investigated before higher level contextual errors.

AN EXAMPLE OF A FAILURE CLASSIFIED AS A WRONG GOAL INFERENCE

```
M-UNSUCCESSFUL-MEDIATION isa M-MEDIATION isa M-PROBLEM
    dispute: orange-dispute-0
    mediation-plan: M-ONE-CUTS-OTHER-CHOOSES
                        expected-contract: M-DIVIDED-OBJ-CONTRACT
    predictions: M-POSSIBLE-EVENTS
                    results-ok:
                       ((*ingest* (actor SISTER1)
                                  (object "sister1's half"))
                        (*ingest* (actor SISTER2)
                                  (object "sister2's half")))
    results-confirmed: nil
    feedback: ((*ingest* (actor SISTER1)
                         (object "sister1's half"))
               (*prepare* (actor SISTER2)
                          (object CAKE1)
                          (inst (*physical-control*
                                    (actor SISTER2)
                                    (object PEEL1)))))
    remediations: nil
    usually-useful-remedies: nil
```




```
M-WRONG-GOAL-INFERENCE isa M-UNSUCCESSFUL-MEDIATION
    dispute: orange-dispute-0
    mediation-plan: M-ONE-CUTS-OTHER-CHOOSES
                        expected-contract: M-DIVIDED-OBJ-CONTRACT
    predictions: M-POSSIBLE-EVENTS
                    results-ok:
                       ((*ingest* (actor SISTER1)
                                  (object "sister1's half"))
                        (*ingest* (actor SISTER2)
                                  (object "sister2's half")))
    results-confirmed: nil
    feedback: ((*ingest* (actor SISTER1)
                         (object "sister1's half"))
               (*prepare* (actor SISTER2)
                          (object CAKE1)
                          (inst (*physical-control*
                                    (actor SISTER2)
                                    (object PEEL1)))))
    remediations: nil
    usually-useful-remedies: (M-USE-ACTUAL-EVENTS M-ASK-ALTERNATE-PARTS
                              M-USE-GOALS-FROM-FEEDBACK  M-ASK-GOALS)
```

Figure 5-15

5.4.2 *Failures due to bad planning*

Another class of failures is due to poor planning. Each of the stages of the planning process that we described in chapter four, and shown again in Figure 5-16, is a potential cause for failure by a problem solver.

REVIEW OF THE OVERALL PLANNING PROCESS

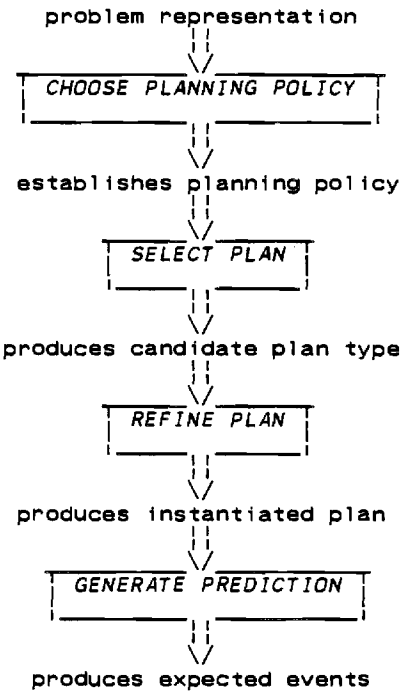


Figure 5-16

As can be seen in Figure 5-16, planning errors can be the result of (1) an incorrect planning policy, (2) selection of an ineffective plan, (3) incorrect plan refinement, or (4) the generation of inappropriate predictions.

We will illustrate planning failures by looking at a failure due to a "wrong planning policy," which is a failure that occurred during meta-planning. Consider the case we call Candy-Dispute-2:

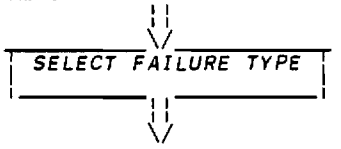
CANDY DISPUTE-2

> A mother is on her way home from the library when she happens on two boys quarreling over a candy bar. She overhears the first little boy shout, "I want it." To which the second boy responds, "So what, I want it too." Unable to resist the opportunity to play mediator, the mother suggests that the boys divide the candy equally between them. Almost in unison, the boys reject the compromise saying, "I want the whole candy bar!" The mother thinks for a minute then suggests that they flip a coin to see who gets the candy. The boys agree and the mother continues homeward.

In Candy-Dispute-2 the children protest the "divide equally" plan because they were unwilling to accept a compromise. They both want to eat the whole candy bar. In terms of our model, this is not a misunderstanding, since the disputants' goals were correctly inferred and the default physical dispute context is appropriate. Nevertheless, the mother-mediator failed to suggest an acceptable plan. This type of failure is caused by a planning error. Specifically, the mother-mediator assumed the default compromise planning policy in the absence of any disconfirming evidence. This planning policy, like any heuristic, works many times, but is not always guaranteed to work. In this particular case, the appropriate remedy is to replan given a new "all or nothing" planning policy instead of a "compromise" one. We illustrate the results of the failure classification process for this in Figure 5-17.

AN EXAMPLE OF BLAME ASSIGNMENT TO A PLANNING FAILURE

```
M-UNSUCCESSFUL-MEDIATION isa M-MEDIATION isa M-PROBLEM
    dispute: candy-dispute-2
    planning-policy: compromise
    mediation-plan: M-ONE-CUTS-OTHER-CHOOSES
                        expected-contract: M-DIVIDED-OBJ-CONTRACT
    predictions: M-POSSIBLE-EVENTS
    results-confirmed: nil
    feedback: ((*mtrans* (actor boy1)
                         (mobject (*goal*
                                  (*or* (*phys-control* (actor boy1)
                                                        (object candy1))
                                        (*phys-control* (actor boy1)
                                                        (object nil))))))
              (*mtrans* (actor boy2)
                         (mobject (*goal*
                                  (*or* (*phys-control* (actor boy2)
                                                        (object candy1))
                                        (*phys-control* (actor boy2)
                                                        (object nil)))))))
    remediations: nil
    usually-useful-remedies: nil
```




```
M-WRONG-PLANNING-POLICY isa M-UNSUCCESSFUL-MEDIATION
    dispute: candy-dispute-2
    planning-policy: compromise
    mediation-plan: M-ONE-CUTS-OTHER-CHOOSES
                        expected-contract: M-DIVIDED-OBJ-CONTRACT
    predictions: M-POSSIBLE-EVENTS
    results-confirmed: nil
    feedback: ((*mtrans* (actor boy1)
                         (mobject (*goal*
                                  (*or* (*phys-control* (actor boy1)
                                                        (object candy1))
                                        (*phys-control* (actor boy1)
                                                        (object nil))))))
              (*mtrans* (actor boy2)
                         (mobject (*goal*
                                  (*or* (*phys-control* (actor boy2)
                                                        (object candy1))
                                        (*phys-control* (actor boy2)
                                                        (object nil)))))))
    remediations: nil
    usually-useful-remedies: (M-USE-ALTERNATE-POLICY)
```

Figure 5-17

5.4.3 *Default failure classification*

The worst case situation that can occur with respect to default failure classification is when there is no advice or feedback to assist in failure classification. In this situation, a problem solver must resort to a methodical examination of its entire line of reasoning. Notice that the problem solving model provides a natural order which can be used to provide explicit direction to default failure classification. This direction is essentially a form of "dependency-directed backtracking" (Doyle, 1979; Stallman and Sussman, 1979) used to control the actions of a problem solver during default failure recovery. Default failure classification can be viewed, in the absence of any direct evidence of error type, as proceeding systematically to investigate, usually by questioning the user, all inferences made between prediction generation (the last stage of problem solving) and problem classification (the first stage). This lengthy static reasoning chain is indicated in Figure 5-18:

BACKTRACKING SEQUENCE IN DEFAULT FAILURE CLASSIFICATION

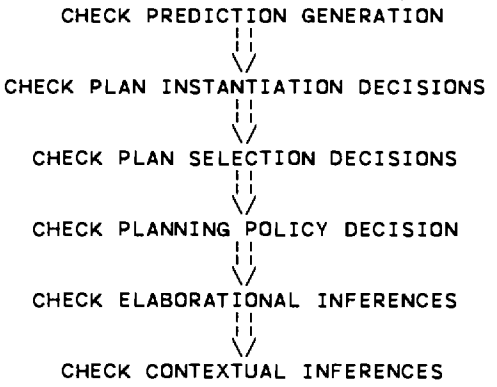


Figure 5-18

Given the long chain of heuristic inferences required in most complex problem solving situations such as dispute mediation, it is easy to imagine how tedious this backtracking process would become (even for a patient client). This long static sequence of default reasoning for failure classification is obviously not a reasonable solution. We see two methods to improve the efficiency of the default failure classification process. The first method uses feedback to direct failure classification. This is useful when there is feedback available, but it does not help a problem solver in its absence. The second method uses case-based reasoning to suggest a plausible failure classification. Case-based reasoning uses feedback when it is available, but can still provide a heuristic choice in its absence. We will discuss this second method in the following subsection. Obviously, when there is neither previous case experience nor feedback, the above static form of backtracking cannot be avoided.

When feedback is available from the environment, it provides the best source for inferring the potential source of error. Our method of default failure classification, when feedback is available, is a matching process that attempts to match the feedback to one of the a priori expectations generated during the final stage of the planning. During the prediction stage of planning, as you may recall, the expected results of successful plan execution are generated (in the MEDIATOR this prediction fills the "results-ok" slot of a M-POSSIBLE-EVENTS frame). At the same time, predictions are generated for events that would be expected if the plan were a success, and those predicted for particular types of failure. When a match occurs on one of these predicted events, then the corresponding type of failure is identified.

In our implementation of this idea, plan predictions fill slots in the "possible events" frame for the class of error that would produce the failure. Thus, an expected error in problem classification would generate a prediction of expected actions that could be used to confirm a "context-error." For example, when the MEDIATOR program generates a M-POSSIBLE-EVENTS frame for a mediation plan, there are slots for "results-ok," "wrong-context," "wrong-goal-inference," etc. (i.e., one slot corresponding to each failure type in our model). Each slot contains a prediction that can be used to "verify" a particular failure type by matching the feedback with these predictions. The prediction that a "wrong-context" error has been made in Candy-Dispute-2 above, for example, might be the feedback that one of the boys "sold" the candy to the other. This would disprove, so to speak, a mediator's inference that the dispute was a "physical dispute" and constitute evidence in favor of a classification as an "economic dispute." This match would result in the failure's classification as a "wrong-context" failure.

In Figure 5-19 below, we illustrate a portion of the failed mediation case corresponding to Candy-Dispute-2. Some of the predictions corresponding to the various failure types are shown filling the appropriate slots in the M-POSSIBLE-EVENTS frame that holds the predictions for this case. For example, the "wrong-planning-policy" slot contains a prediction that represents the abstract concept "all-or-nothing." The feedback from the boys, which is shown in Figure 5-19 opposite the slot "feedback" is a conceptual dependency form that represents the boys' response that they both want all the candy (understood as "all-or-nothing"). The match process essentially chooses between the alternate classification possibilities based on a best fit between the feedback and these predictions. In the Candy-Dispute-2, this results in a classification of the failure as a "wrong-planning-policy." This decision is shown at the bottom of Figure 5-19.

MATCHING FEEDBACK WITH PREDICTIONS FOR DEFAULT FAILURE CLASSIFICATION

```
dispute: candy-dispute-2
predictions: M-POSSIBLE-EVENTS
                results-ok:
                     ((*ingest* (actor boy1) (object (half candy1))
                      (*ingest* (actor boy2) (object (half candy1)))
                wrong-context:
                     (*atrans* (seller &actor1)
                               (object (&object (value &value)))
                               (buyer &actor2))
                wrong-goal-inference:
                     (&action (actor (*or* boy1 boy2)))
                       where &action not *ingest*
                wrong-planning-policy:
                     (*or* (&action (actor &actor) (object &object))
                           (&action (actor &actor) (object nil)))
                :
                :
feedback: ((*mtrans* (actor boy1)
                     (mobject (*goal*
                                   (*or* (*phys-control* (actor boy1)
                                                         (object candy1))
                                         (*phys-control* (actor boy1)
                                                         (object nil)))))))
          (*mtrans* (actor boy2)
                    (mobject (*goal*
                                   (*or* (*phys-control* (actor boy2)
                                                         (object candy1))
                                         (*phys-control* (actor boy2)
                                                         (object nil)))))))
```

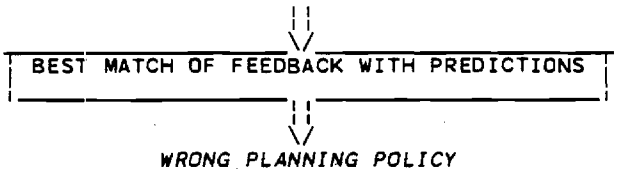


Figure 5-19

We recognize several problems in this default classification scheme. The first, discussed in section 5.3.2, is another instance of the matching problem, this time between feedback and the various types of failure predictions. Second, we now must decide in what order we will examine the predictions and whether we want a "best first" or "exhaustive testing" of all the different predictions. One way to handle this control problem is to hierarchically arrange the failure types in a default preference order (similar to our ordering of problem classes for the dispute domain). The sequence of match testing in this scheme can be either by the estimated "fallibility" of the inference based on some a priori knowledge,* an estimate based on failure experience, or in reverse order of the inference dependencies shown in Figure 5-18 (this is because lower level failures do not rule out higher level failures). Of all these methods, we prefer the use of previous failure experience to direct the sequence of match testing. However, the current implementation of the MEDIATOR uses a fixed sequence in reverse order of that shown in Figure 5-18.

Our technique for failure classification by default reasoning, as described above, seems to capture a capability of problem solvers that has been described as "what-will-happen-if" reasoning (Chandrasekaran, 1983). This type of reasoning is most evident in expert diagnosticians as they evaluate alternative hypotheses that explain a situation. For example, "What would happen if valve A is closed and pressure continues to build?"; or "What would happen if I inferred the disputants' goals incorrectly?" To us this is exactly the sort of reasoning that is used to generate predictions during planning and during failure classification. We have not explored this type of reasoning to any great detail except as described above. One possibility is that failure experiences can be used to provide the predictions that are generated during the planning process. This would allow modeling of the acquisition of "what-will-happen-if" knowledge and replace the static prediction generation process described above.

---

*This is the method suggested by O'Rorke (1983) for failure classification in a natural language understanding system.

5.4.4 Case-based failure understanding

Case-based reasoning offers the possibility of avoiding both the tedious backtracking and default failure classification processes described above. When a failure causes the recall of a previous failure case, the cause of the failure in the recalled case is examined to see if it can explain the current failure. We transfer the failure classification from the recalled case unless there is evidence of contradiction with other domain knowledge in the case. Suppose for example, that the recalled failure was a "wrong goal inference." If the goals of the current case were inferred, then that failure classification is transferred. If, on the other hand, the goals of the current case were given explicitly in the problem description, "wrong goal inference" would be ruled out. When there is feedback available, the transfer is ruled out only if the feedback does not match (optimistically) the prediction provided by that failure type. Thus, we bias our problem solver to suspect failures that had been previously recognized. This seems to work well for dispute mediation.*

Our case-based algorithm is shown in Figure 5-20.

A CASE-BASED ALGORITHM FOR FAILURE CLASSIFICATION

1. Recall similar cases of failure and select the one failure most similar to the current case.

2. If there is a similar case, then check the applicability of the same classification by matching the feedback with the prediction corresponding to that classification. If the classification is not ruled out, then transfer that classification and reformulate the failure as an instance of this failure type.

3. Otherwise, classify the failure by default reasoning.

Figure 5-20

In the Sinai dispute case, for example, the problem solver generated an incorrect prediction that Israel and Egypt would "divide the Sinai equally." This failure causes the mother to remember another time when the "divide equally" plan (one cuts, the other chooses) also failed. The recalled case, in that instance, was classified as being caused by a "wrong goal inference." Because there is no reason why this classification cannot apply here, the current case is reinstantiated as a failure of type "wrong goal inference." This is the behavior illustrated in the sample output from the MEDIATOR program shown in Figure 5-21.

---

*This is an extension of the idea that a problem solver who once succeeds in solving a problem will tend to use the same methods again in a similar situation (i.e., set effects) (Luchins, 1942). In this situation, instead of selecting a planning method, a problem solver selects a failure classification.

I/O BEHAVIOR DURING CASE-BASED FAILURE CLASSIFICATION

```
(mediator sinai-dispute t)
Considering the following problem:
   Israel and Egypt both want the Sinai,
   which has been presented as ako M-DISPUTE.
      :
      :
I suggest that the plan called "one cuts the other chooses" be used.
Do you agree, that this is the best solution? (Y or N) No.
   **** "one cuts the other chooses" not acceptable ****

Can you provide any comments that might help me remedy this failure?

((*MTRANS* (ACTOR ISRAEL)
           (MOBJECT (*GOAL* (*NAT-SECURITY* (ACTOR ISRAEL)
                                            (OBJECT SINAI)))))
 (*MTRANS* (ACTOR EGYPT)
           (MOBJECT (*GOAL* (*NAT-INTEGRITY* (ACTOR EGYPT)
                                             (OBJECT SINAI))))))
Attempting to explain this failure and find a new solution.

Considering the following problem:
   failed mediation for Israel and Egypt both want the Sinai,
   which has been presented as ako M-UNSUCCESSFUL-MEDIATION.
   It will be referred to as #<M-UNSUCCESSFUL-MEDIATION 40544074>
ATTEMPTING TO RECALL SIMILAR PROBLEMS IN ORDER TO CLASSIFY THIS ONE...

ATTEMPTING TO RECALL SIMILAR FAILURES ...
  looking for previous mediation plan failures...
  looking for failures with similar disputants or with similar goals..
  looking for failures involving similar objects...
   reminded of the failed mediation for two sisters are quarreling over
      an orange for which the plan "one cuts the other chooses" also
      failed and because the object in that case, ORANGE1, and SINAI are
      both of type M-PHYS-OBJ.
There was one previous case found.
   #<M-WRONG-GOAL-INFERENCE 5304703> was the
   failed mediation for two sisters are quarreling over an orange.
Failure in that case was because of M-WRONG-GOAL-INFERENCE.
Transferring that classification to this failure.
The current failure will be referred to as
   #<M-WRONG-GOAL-INFERENCE 40544535>
```

Figure 5-21

5.5 *Failure remediation*

Using the representation of the failure developed during the understanding phase, a known remedy associated with the specific class of failure must be selected and applied in order to change (i.e., remedy) the knowledge that caused the problem solver's faulty reasoning. This is the stage of recovery from failure that we call failure remediation. It corresponds to the planning and execution stages of our problem solving model that was explained in chapter four. By analogy to the planning process, remedies correspond to plans. They provide canned approaches associated with failures of different types just as plans are associated with specific problem types in the problem domain (e.g., try "one cuts, the other chooses" first for physical disputes over possession of food).

Like the planning process described earlier, this stage of failure recovery involves subprocesses devoted to selecting the remediation policy (i.e., meta-planning for remediation), selection of an abstract remedy, refinement and instantiation of the chosen remedy, and generation of predictions based on the remedy's application. In addition to the planning process previously described, failure remediation also involves the actual execution of the remedy. This additional step was not included in the planning model described in chapter four because we assumed that the planner was operating in the advisory role and not in an executor capacity. During failure recovery, however, a problem solver must execute the remedy so that the appropriate change in its own reasoning may be effected. It is for this reason that execution is included in failure recovery.

Remedies, like plans, have preconditions that determine their applicability and are organized hierarchically from the abstract to the specific as indicated by Figure 5-22. There are two basic types of remedies: those that are used to remedy misunderstanding errors and those that address planning errors. Each remedy type is useful for repairing the problem solving model. For example, one remedy for repairing a goal misunderstanding is to infer an

alternate goal for a disputant by analysis of that disputant's actions after the mediation plan is executed (i.e., M-USE-ACTUAL-RESULTS). This is the remedy used by the mother-mediator to remedy her reasoning in Orange-Dispute-0. A portion of the MEDIATOR's generalization hierarchy for remedies is shown in Figure 5-21.

A PORTION OF THE REMEDY ABSTRACTION HIERARCHY

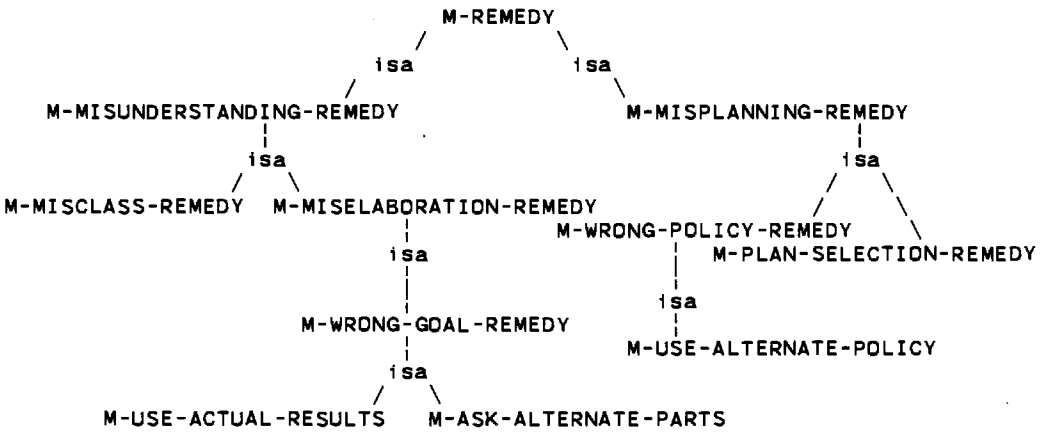


Figure 5-22

During remedy selection, as in plan selection, a problem solver makes specialization decisions that transform the general "remedy" concept into a specific remedy instance. For example, a general remedy for "failures caused by understanding errors," is called M-WRONG-GOAL-REMEDY, but this abstract remedy requires specialization (e.g., as either M-USE-ACTUAL-RESULTS or M-ASK-ALTERNATE-PARTS) before it can be useful to a problem solver. We will concentrate only on specific remedies in our discussion since they perform the most interesting work during remediation. Most specific remedies, once selected, can be applied directly to affect the problem solver's reasoning. For example, if the MEDIATOR believes that its representation of a disputant's goal is in error, it changes that specific portion of its internal representation. Other specific remedies, however, direct the problem solver to investigate the environment for information (e.g., "ask the disputants if they want x"), which eventually also leads to some internal change in the problem solver's knowledge. This and other specific remedies will be discussed in the following sections.

Case-based reasoning can be used during failure remediation to suggest the remedy that was associated with a previous similar failure. When the preconditions for that remedy are satisfied, it is transferred and employed to change the problem solver's reasoning. This process is analogous to the use of case-based reasoning in planning. We will illustrate this in a later subsection.

5.5.1 Remedies for misunderstandings

As we discussed in section 5.4.1, there are two major sources of misunderstanding errors: problem misclassification and erroneous elaboration of the representation. Erroneous elaboration of problem details, in particular the goals of another agent, seems to be the most common source of failure (Fisher and Ury, 1981; O'Rorke, 1983). We will focus our discussion on remedies for this small subset of possible misunderstandings. This will provide details on the parts of our methodology that are not completely analogous to the planning process discussed in chapter four. In particular, we will present structured algorithms for some specific remedies and provide illustrations in some cases from the MEDIATOR program. The specific remedies for goal misunderstanding errors that we have identified are the following:

1. use actual events from plan execution to infer goals
2. ask about alternate parts of object to infer goals
3. use goals directly from feedback
4. ask about other known uses of object
5. consider other themes to infer goals
6. ask for goals directly

Some of these remedies use available information only (1 and 3) and therefore require that that information is already available (e.g., recognized goals or actual results included in the feedback). Others (2 and 4) represent remedies that depend on particular knowledge (e.g., knowing parts of objects or other uses for objects). Still others (e.g., 5 and 6) represent alternate approaches to finding reasonable goal inferences. For example, we can look for other themes that can be used to provide goal inferences indirectly as suggested by remedy 5. Or, on the other hand, remedy 6 takes the direct approach and asks explicitly for

the goals. You will recall that this was the remedy used by the librarian in the window dispute example at the beginning of this chapter. We will describe some of these in more detail below.

Remediation from goal inference failures depends on identifying the source of the inference. For example, when the actor's goals are inferred from the uses of a disputed object and the object has parts that can be used for different purposes, then reasonable alternatives for goals include uses associated with parts of the disputed object. This is the remedy known as "ask alternate parts" and it depends on the problem solver having the knowledge of how to decompose the disputed object into parts. The general structure of this remedy is shown in Figure 5-23:

"ASK ALTERNATE PARTS" REMEDY FOR WRONG GOAL INFERENCE

```
IF Gi was inferred from a disputed
   object with parts (P1, P2, ...)
   and some parts have normal uses
   (e.g., G2, G3 ...) different from Gi
THEN consider G2, ... as alternate goals
     and ask disputant for confirmation.
```

Figure 5-23

This is the remedy that was used by the MEDIATOR to correct its goal inference in an earlier implementation of the program when it was faced with using default reasoning in an initial attempt to resolve Orange-Dispute-0 without using case-based reasoning. The program has been given a priori knowledge that pieces of fruit have four parts: seeds, fruit, juice, and peel. It also knows that the fruit and juice of fruits are both used for "ingest" purposes, while the seeds are used for "growing" and the peel can be used for "preparing other food." It also knows that an orange is a fruit and that orange1 is an instance of an orange. Using this knowledge, it avoids asking the client about parts of the orange that are used for ingestion, since that is the failed goal that is being repaired. This is illustrated below.

I/O BEHAVIOR OF THE MEDIATOR USING THE "ASK ALTERNATE PARTS" REMEDY

```
I suggest that the plan called one cuts the other chooses be used.
Do you agree, that this is the best solution? (Y or N) NO.
   **** FAILURE of one cuts the other chooses ****
Attempting to explain this failure and find a new solution.
     :
     :
Looking at the remedy called "ask alternate parts"
   which appears applicable.

Do you think SISTER1 is really interested in the seeds from orange1? NO.

Do you think SISTER1 is really interested in the peel from orange1? NO.

My previous goal inference for SISTER1 will be retained for now.
Another possibility I know about is that SISTER2
  wants a part of orange1.

Do you think SISTER2 is really interested in the seeds from orange1? NO.

Do you think SISTER2 is really interested in the peel from orange1? YES.

SISTER2 is now represented as having the goal #<M-PREPARE 22477535>

(*GOAL* (*PREPARE* (ACTOR SISTER2)
                   (OBJECT FOOD1)
                   (INST (*GOAL* (*PHYS-CONTROL* (ACTOR SISTER2)
                                                 (OBJECT PEEL1))))))
Remediation complete.

Given this new information, I'll reconsider this problem.
     :
     :
```

Figure 5-24

Another remedy used to recover from goal inference failures is called "ask about other known disputant uses". This remedy depends on a problem solver having some knowledge of alternate uses for the disputed object by the specific disputant or class of disputants. For example, a problem solver may have only limited direct knowledge about the goals of a polity such as Egypt in the Sinai dispute. So an error in inferring Egypt's goal is quite likely. But if a problem solver knows something about polities in general, then failure recovery can proceed from this knowledge. In this case, knowing that land is used by polities in support of "national security" and "national integrity" goals provides information useful for directing the investigation and repair of goal inference failures. The structure of this plan is shown in Figure 5-25 below:

"ASK ABOUT OTHER KNOWN OBJECT USES" REMEDY FOR WRONG GOAL INFERENCE

If goal G1 was inferred from the
normative use of the disputed object
by the disputant and there exists other
goals G2, G3 ... associated with the
disputant's use of the disputed object
Then consider G2, G3 ... in turn as the
new goal and ask disputant for confirmation.

Figure 5-25

This remedy was also employed in an earlier version of the MEDIATOR which was used to remedy its reasoning in the Sinai dispute. A portion of this behavior is illustrated in Figure 5-26.

I/O BEHAVIOR OF THE MEDIATOR USING "ASK ABOUT OTHER OBJECT USES"

I suggest that the plan called one cuts the other chooses be used.
Do you agree, that this is the best solution? (Y or N) *NO.*
**** FAILURE of one cuts the other chooses ****
Attempting to explain this failure and find a new solution.
:
:
Looking at the remedy called "ask about other object uses"
which appears applicable.

Do you think ISRAEL really has M-NATIONAL-INTEGRITY goals? *NO.*

Do you think ISRAEL really has M-NATIONAL-SECURITY goals? *YES.*

Another possibility I know about is that EGYPT intends
to use the Sinai differently.

Do you think EGYPT really has M-NATIONAL-INTEGRITY goals? *YES.*

Given this new information, I'll reconsider this problem.

Figure 5-26

Since goals are derived from themes (Schank and Abelson, 1977), another source of goal error is associated with incorrect theme inference. If the goal was inferred because of a thematic relationship, then recovery will normally be directed toward the examination of the theme derivation or alternative goals enabled by that theme. The structure of this remedy is as shown in Figure 5-27:

"CONSIDER OTHER THEMES" REMEDY FOR WRONG GOAL INFERENCE

```
If the goal Gi was inferred from either a
   role theme or an interpersonal theme
Then
     IF the theme was inferred (i.e., not given)
     Then consider other theme inferences
        consistent with the original problem
        representation
     Else when alternative themes are also
          inconsistent remove all theme inferences
Else IF the theme was a given part of the initial
    representation consider other goals enabled
    by that theme.
```

Figure 5-27

5.5.2 *Remedies for planning errors*

Planning errors, in our model, can be of four general types: selection of an incorrect planning policy, selection of an incorrect plan, incorrect plan refinement or instantiation, and generation of improper predictions. We have not investigated this portion of our model extensively because we believe it to be analogous to the process discussed in the previous section. In particular, we have not implemented any remedies for planning failures in the MEDIATOR program. We have implemented an ad hoc method of selecting another plan when told to do so, but this was done in order to investigate inductive learning of plan preconditions (see section 4.7). We will, therefore, only mention a few remedies that have been considered to date and present a structured algorithm for only one remedy in order to illustrate this part of the process. Specific remedies for planning errors that we have considered are the following:

1. use plan directly from feedback
2. eliminate plan, select another
3. use planning policy directly from feedback
4. ask about alternate planning policy

The first two remedies are concerned with failures attributed to the selection of the wrong plan. With a static set of a priori preconditions for each plan, we normally do not expect problem solvers to fail because they select the wrong plan (especially if we have done our job of constructing appropriate preconditions). However, when a novice problem solver is in the mode of learning plan preconditions, we expect that there will be many instances of plan selection failure. In this situation, we anticipate needing remedies of the sort listed above. Remedies 3 and 4 are useful for planning policy errors (i.e., meta-planning errors). We envision using such remedies to direct the selection of alternate policies. The nature of these remedies is illustrated in Figure 5-28:

"ASK ALTERNATE POLICY" REMEDY FOR PLANNING POLICY ERRORS

```
IF planning policy P1 was inferred by
   default or case-based reasoning and
   alternate policies (P1, P2, ...)
   have not failed in previous attempts
   to solve the current problem
THEN consider P2, ... as alternate policies
     and ask disputant for confirmation.
```

Figure 5-28

5.5.3 *Case-based remediation*

Because this stage of remediation is analogous to the planning stage of our problem solving model, we employ case-based reasoning in exactly the same ways as we described in chapter four. If a similar failure is recalled, not only can this failure possibly supply the explanation for the failure, but it can also suggest a remedy for the error. For example, Orange-Dispute-0 provides a negative exemplar for the "one cuts the other chooses" plan. Retrieval of that failed experience provides the plausible explanation that the failure was caused by an incorrect goal inference and the suggested general remedy "use the actual events from plan execution to infer goals" since that was the way the mother-mediator resolved her misunderstanding.

Case-based reasoning functions in the planning stage of remediation just as in previous situations. We illustrate this by way of the remedy selection process which is described via the algorithm shown in Figure 5-29.

A CASE-BASED ALGORITHM FOR REMEDY SELECTION

1. If a previous failure is already known, then using that exemplar, go to step 2, otherwise try to retrieve a previous similar failure and then go to step 2.

2. If the preconditions for the remedy used in the recalled failure are satisfied, then transfer and select that same remedy for the current failure.

3. Otherwise, select the remedy by default reasoning.

Figure 5-29

In remedy selection, the preconditions for each remedy serve to constrain the transfer of a recalled remedy in the same way that a recalled plan is constrained during the planning process.

5.6 Some implications

5.6.1 Learning from failure

Problem solvers need to evaluate their decision-making performance in order to learn from their experience. Such an evaluation is necessary in any system that is going to reliably use its past experience to solve a new problem. Evaluation of success or failure biases the problem solver in the future. If a case is judged a success, then future similar cases will be resolved using this case as a positive exemplar. If a case is judged a failure, then more effort is required to satisfactorily resolve it and it will tend to be recalled later only as a negative exemplar.

In our model, learning occurs as a by-product of integrating each case into long-term memory. Positive exemplars allow a kind of rote learning about the circumstances that existed when success was achieved. These cases are useful, as we have shown, in helping resolve future problems more efficiently by providing specific guidance in specialized circumstances. As more success is achieved, case-based problem solvers generalize from the specific features of case instances so that their memories, at the highest level of abstraction, eventually describe the conceptual "space" of component features associated with successful efforts (Mitchell et al., 1983). Successful problem solving thus permits the learning of only that part of the problem domain that gives rise to confident predictions of success. When a new case has features that fall within the conceptual space of previous successful cases, we say a problem solver has confidence based on experience. Such a positive measure of confidence based on experience is not possible with the usual rule-based approaches. It is also very different from the usual measures of confidence based on certainty factors supplied either a priori or by a knowledgeable user (e.g., Shortliffe and Buchanan, 1975; Zadeh, 1965).

Negative exemplars also allow rote learning, but the difference is that the learning encompasses those circumstances that existed when failures occurred. At a minimum, the recording of failures allows a problem solver access to them for potential use in resolving new failures. In the most favorable of circumstances, the integration of failure cases into memory allows a problem solver to abstractly describe the conceptual feature space associated with failures so that they may possibly be used as evidence to avoid failure before the fact. To the extent that the space associated with success is disjoint from the space associated with failure, a problem solver can estimate his ability to deal with a new problem (e.g., whether a known plan is applicable) depending on whether the various features of the new case correspond to features within the space described by failure instances. In addition, when the spaces corresponding to successes and failures precisely partition a problem solver's knowledge of the domain, then we can say that the problem solver has become an "expert." This ability to refine the problem domain more precisely comes about only through failure. If problem solvers only record successes, then a significant source of predictive knowledge is missing.

5.6.2 Top level control of problem solving

Because we deal explicitly with failure in our problem solving model, we have had to face two additional issues: recording the problem solver's reasoning during failure recovery and coordinating the multiple instances of the problem solving model that may be active at different times. We need to record and reason about the problem solver's inferences so that candidate sources of error can be investigated. Without such a mechanism, reconsideration of previous inferences, such as discussed in this chapter, becomes impossible.

One highly regarded technique for failure recovery is dependency-directed backtracking (Stallman and Sussman, 1979; Doyle, 1979). In dependency-directed backtracking, a new assertion includes a dependency record indicating the sequence of facts that led to the current belief. These records are usually implemented as a simple list of antecedent lists (one for each way a fact was deduced) and consequence list (those facts derived from this fact). In systems that employ dependency-directed backtracking such as the system for doing electric circuit analysis called EL (Stallman and Sussman, 1979), there has been no attempt to incorporate this type of problem solving experience into a conceptual memory model. The consequence is that even though the program may recover from its error and even learn that a sequence of reasoning should be avoided in resolving the current problem as EL does, it will commit the same error if given the same problem again in the near future.

Instead of a list of dependency records attached to the inference itself, which makes the detection of error patterns a more difficult process, we use a globally accessible blackboard (Erman, et al., 1980; Hayes-Roth, 1983) to record and organize inferences made during problem solving. Since this information concerns the problem solver's reasoning about its own reasoning, the blackboard, which is external to the representation of the case, is the natural location for this sort of meta-knowledge. There are five elements to our blackboard implementation:

1. the problem solving executive element
2. the problem representation element
3. the solution plan element
4. the problem solver's long term memory element
5. the currently active inference element

The executive element provides for the scheduling of problem solving processes according to a model of problem solving behavior. For example, the default sequence of processes is understand the problem, suggest a plan for its solution, test results, and follow-up failures. This is our basic problem solving model.

The problem representation element records the initial representation of problems and maintains the current representation as understanding processes reformulate and elaborate it. The solution plan element records the current planning policy, the current plan, previous plans attempted, expected results, and actual results. The long term memory element provides the top level entry into the problem solver's episodic data base. We will discuss the organization of this component in chapter six.

The currently active inference element has an area associated with each phase in the problem solving model (e.g., problem classification, plan selection, etc.). As inferences are made during each phase of problem solving, they are recorded in the appropriate area of this element. Failure recovery processing is directed, during failure classification, toward those inferences which are appropriate given the specific nature of the current failure and the information provided from feedback (e.g., investigate inferences dealing with goal elaboration).

Processing control, during failure recovery, propagates outside one specific inference area based on the implicit relationships among the different processes as discussed in section 5.4.3 earlier. In this way, a failure classified as a planning policy failure will bypass the goal and classification inference areas and begin investigating only those inferences within the planning policy area of the blackboard. The blackboard structure also provides a convention for incorporating experience into the failure recovery process. When a failure is finally classified and recovery completed, the features of the failure as well as the understanding and remedy become components of the problem solving case. These pieces of information come from the blackboard. They are included with the case when it is integrated into memory. The features of this case as well as previous failure cases become available from memory when a similar failure context triggers its recall.

Because we initiate a separate instance of the problem solving model to deal with failure recovery, there is a danger that the problem solver will "get into a high level loop" and oscillate between equally unproductive lines of reasoning. For example, let us say a problem solver has several plans available to deal with the problem. One plan, P1, leads to a failure. During failure recovery, the plan is identified as the cause and plan reselection is indicated. Whereupon a second plan, P2, is selected and also fails. Once again, during failure recovery, plan reselection is directed. If the problem solver does not know about its previous decisions, it could choose plan P1 again, leading to a potentially infinite loop between plans P1 and P2. Because the blackboard is available as a "working memory," the problem solver can keep track of previous decisions, such as what plans have been tried, by looking at the appropriate portion of the blackboard, thus avoiding this kind of high level looping.

5.7 Summary

This chapter has been concerned with failure recovery, a facet of problem solving that most AI problem solving systems avoid. Our approach to failure recovery is based on the recursive application of the problem solving process on itself. A problem solver initiates failure recovery when failure is discovered as a result of feedback. We have operationalized this as a matching process that attempts to categorize feedback in terms of known successful

representations and known categories of failure. When failure is detected, the problem understanding process is instantiated to further classify and elaborate the failure representation. During failure recovery, this activity is the blame assignment process. After understanding the failure, the planning process next selects, refines, and executes a known remedy for the failure. Remedies for errors in reasoning correspond to domain problem solving plans and are another source of problem solving (introspective) knowledge. Once a problem solver's reasoning has been corrected, the process is reattempted for the original problem.

Our problem solving model provides natural categories of failure. At an abstract level, we can specify failures as understanding, planning, or testing failures corresponding to each of the major processes in our model. On a finer grain, we can specify that understanding failures are the result of misclassification or miselaboration. Planning failures result from bad policy decisions, misselection of plans, poor refinement, or bad prediction. Failures during the evaluation phase can be caused by the false recognition of success or the false belief that a failure has occurred. With failure so classified, specific remedies can be chosen to address each type of failure.

Case-based reasoning applies in this instance of problem solving just as in our original presentation. Previous failures in reasoning, like previous domain cases, can be used to assist in failure understanding (blame assignment) and remedy selection (correcting the failure). We have provided algorithms and illustrative examples of these processes from the MEDIATOR program.

One of the important consequences of our decision to recursively apply the problem solving model is the necessity of recording the problem solver's inferences, so that they can be investigated during blame assignment and remediation. This necessitates additional types of generalized memory structures in a problem solver's long term memory to accommodate these additional concepts. We have adopted the blackboard construct as a top level mechanism within which to record a problem solver's inferences and to control the problem solving model. This prevents high level looping between alternatively bad choices.

CHAPTER VI

A CONCEPTUAL MEMORY FOR CASE-BASED PROBLEM SOLVING

> "In order to obtain the solution, we have to extract relevant elements from our memory, we have to mobilize the pertinent parts of our dormant knowledge. We cannot know, of course, in advance which parts of our knowledge may be relevant; but there are certain possibilities which we should not fail to explore. Thus, any feature of the present problem that played a role in the solution of some other problem may play again a role. Therefore, if any feature of the present problem strikes us as possibly important, we try to recognize it. What is it? Is it familiar to you? *Have you seen it before?*" (Polya, 1945)

6.1 Introduction

When novices are asked repeatedly to solve problems, we know that initial erratic performance is soon replaced by a steady, often mechanistic, performance (e.g., Luchins, 1942). It seems that novices use their previous successes and failures to refine and guide their actions. In other words, problem solvers *remember past cases and are guided by the results of their past actions* (e.g., Reed and Johnsen, 1977; Ross, 1982). In our model of case-based reasoning in problem solving, we provide a computational explanation for this process. In the preceeding three chapters, we have detailed a process model of problem solving that uses previous case experience to guide decision making. For example, successfully resolving one dispute leads the MEDIATOR program to make similar decisions in the future; conversely failure discourages the program from making the same decisions when faced with similar situations. For case-based reasoning to represent a viable problem solving approach, it follows that computer programs must be able to recall appropriate cases from a long-term cache (Lenat et al., 1979) of case experiences.

This chapter details the specifications for a conceptual long-term memory (Kolodner, 1984) which supports case-based problem solving. This memory supplies not only a cache of previous problem solving cases, but also organizes these cases in such a way that the interaction of memory with case-based problem solving reflects the additional knowledge gained from these experiences. We refer to our memory organization as a conceptual memory because its case knowledge is hierarchically-organized around important concepts in the problem domain. In chapter two, we specified the important concepts in the dispute mediation domain. These concepts are represented as primitive concepts in an internal language used to model the problem domain. Important problems in the design of a conceptual memory for a case-based problem solver include: (1) providing an operational means of determining the similarity between any two domain concepts, (2) providing a means of incrementally building a memory of cases to support case-based reasoning, and (3) providing a retrieval mechanism that produces the appropriate cases needed to support the problem solver. In the following sections, we will address each of these three issues in some detail.

In previous chapters, we provided an overview section which served to introduce our ideas from a broader perspective. We will deviate from this pattern in this chapter, since an overview was effectively presented in chapter one, section 1.4. Those readers not familiar with the basic ideas of a "dynamic memory" (Schank, 1982) or a "conceptual long-term memory for events" (Kolodner, 1984) are invited to review section 1.4 before reading this chapter.

6.2 Long-term memory requirements for case-based reasoning

Our specific model of problem solving, detailed in chapters three through five, imposes several functional and performance requirements on a long-term memory of cases. In this section, we will explicitly identify these reqirements as motivation for specific design decisions reflected in our current implementation.

6.2.1 Some functional requirements

There are four functional requirements on conceptual memory implied by our model of case-based reasoning.

1. the ability to retrieve previous case exemplars based on a minimal description of a problem

2. the ability to retrieve abstract cases of success and failure under specific situations

3. the ability to store cases such that they are retrievable based only on their similarity to new problems

4. the ability to identify the most applicable case from a set of potentially applicable cases

First, our conceptual memory must provide the capability to retrieve previous cases based on a very brief or "sketchy" description of the problem. This requirement is dictated by the fact that real world problems rarely come completely specified. This is the reason our problem solving model includes classification and elaboration phases during understanding. However partial a problem description is, if a problem solver can be reminded of previous cases similar to the current case, it may gain a better understanding of the problem and have an easier time solving the problem. The combination of this capability and the understanding process provides the initial direction to a case-based problem solver's reasoning.

Second, we need to be able to retrieve generalized cases of successful and unsuccessful problem solving in order to support any inductive learning required of a problem solver. Without such a capability, a novice problem solver might be forced to remember and compare all training instances each time a course of action is being considered. Such a scheme would be necessary to insure that a problem solver's actions were consistent with previous case instances. Generalized successes allow a problem solver to incrementally expand its knowledge of some problem types and encourages the selection of some actions over others. Generalized failures serve to incrementally restrict the types of problems for which other actions are tried (e.g., Mitchell et al., 1983).

Third, the organization of conceptual memory must evolve in such a way that the retrieval process is not impaired as new cases are added. This is necessary to insure that new cases have the opportunity to influence later problem solving. This capability makes a problem solver more responsive to changes in the environment of the problem domain. For example, a novel case once added to memory should be able to help in later problem solving no matter how much unrelated problem solving has gone on in between. In addition, any new case has the potential to add to a problem solver's generalized domain knowledge. New generalizations serve to relate problems that might seem unrelated on the surface (e.g., the Sinai dispute and Orange-Dispute-0). The interaction of a problem solver with an evolving memory provides the self-adaptive capability that we desire in computer systems.

Fourth, we must be able to identify the most applicable cases from the possibly thousands of cases that may be stored in a memory. This requirement derives from the fact that, on the one hand we need memory to retrieve any potentially applicable cases to increase the chances of locating good exemplars, but on the other we want to focus on only those cases that seem most heuristically useful in the current situation. As we have shown in our case-based algorithms, such a capability is a necessary first step to making useful information available for transfer. If we identify the wrong previous case, either no information or the wrong information will be transferred. This task is complicated by the fact that a problem solver will rarely see exactly the same problem twice.

6.2.2 Some performance requirements

Besides these four functional requirements, our problem solving model imposes a set of general performance requirements on a long-term memory of cases. These performance requirements are relative to problem solving models without a conceptual memory.

1. A case-based problem solver must perform "better" than a comparable problem solver without a long-term memory

2. Performance of the processes used for retrieval and selection of the appropriate cases must be relatively efficient. Retrieval performance cannot slow appreciably as more cases are added to long-term memory.

3. performance of memory update processes can be traded off to improve retrieval performance

---

*Four of these relationships were identified by Wickelgren (1974) as existing between any two problems. Certainly a "concept" is more general than a "problem," but I'm essentially saying the same thing. Where I differ with Wickelgren is that his fifth relationship was called "similar" which he describes as being "partially analogous;" while my fifth relationship is the "sibling" relationship (i.e., A and B have a conceptual parent in common). Unlike Wikelgren, I view all these relationships as "various degrees of similarity." This is closer to the ideas of Gentner (1982), for example.

To be useful, case-based reasoning must offer a performance improvement to other problem solving approaches. It is in this sense, that we say it must be "better." Of course, what is meant by "better" is open to interpretation. For example, is it "better" if a problem solver automatically adjusts its reasoning as its problem environment changes even at the cost of longer processing time? Is it "better" if a human programmer is required to modify a problem solver's reasoning when a failure occurs or if the program must spend considerable time repairing itself? Is it "better" to have a problem solver with only a single line of reasoning or to spend the processing time and space to build a conceptual memory of cases? These questions cannot be answered in general, but for problems which involve long lines of reasoning and which invclve decision making using data that is relatively static,* case-based reasoning is better in that long computational decisions can occasionally be avoided.

When analyzing the potential performance of a case-based problem solver, the most obvious impact on performance is made by memory retrieval and case selection processes. This is our second performance requirement. Memory performance, at least during retrieval, can not slow down appreciably as more cases are added to conceptual memory. This requirement is again dictated by the fact that the case-based approach, to remain a viable alternative to static problem solvers, must perform competitively after thousands of case experiences. Realistically, we expect that after an inital learning period, a case-based problem solver will most likely settle into a steady knowledge state. Presuming that over a period of use, a problem solver has dealt extensively with the full range of problems in its domain, we suspect that very little new knowledge will be added to conceptual memory. At this point, we expect even a case-based problem solver to appear static since it will have reached a plateau of expertise. It's value, at this point of maturity, would be in retaining an ability to respond to changes in the problem environment. This capability would be initiated as a result of a failed problem solving experience.

Recognizing the importance of retrieval performance, we also realize that cases can be retrieved efficiently only when they have been stored in a manner that supports the retrieval process, this is our third performance requirement. The overall organization of cases must ensure that the most similar cases are identified quickly and made available. Because we can not know in advance what form the problem description will take, we need multiple indices for cases that include all features believed important to the recognition of similarities. In addition, the memory process responsible for index selection during retrieval must perform exactly the same function during memory update. This provides the most opportunity for locating similar cases by effectively looking in memory at the place where the current case would be located if it were there. In some domains, problem solving performance is critical only from the time a problem is presented until a potential solution is returned (e.g., a physician's time during diagnosis or locating the potential source of a toxic chemical spill). In these situations, the time spent updating a conceptual memory after a case has been resolved is relatively unimportant. By trading off additional space for multiple indices and additional processing time during update to maintain memory's organization, we expect to improve overall retrieval performance.

6.2.3 The requirement for similarity

In order to fulfil the requirements presented above, we will need to define memory processes, structures, and put forth a definition of similarity that allows the problem solver to recognize the applicability of previous cases cases to the one at hand. Consider the following version of Orange-Dispute-0, which illustrates the need for such a definition.

> A mother comes home from the library to find her daughters quarreling over an orange. Immediately she is reminded of two similar events that ocurred earlier that day. First, she recalled a squabble that two men had over opening a window in the library. Second, she recalled her encounter with two little boys fighting over a candy bar. Recognizing the similarity between her daughters' quarrel and the little boys' fight, she decides to consider it further. She reasons that a good solution to her daughters' quarrel would be to divide the orange between them; analogous to her earlier successful suggestion to the little boys that they should divide the candy.

---

*The normal use of food is a static piece of knowledge, but the state of a traffic signal is usually dynamic knowledge.

While the concept "orange" and the concept "candy" are lexically dissimilar, conceptually, they are very close. Both are specialized concepts of the more general concept "food." It is reasoning that is based on this type of *conceptual similarity* that needs to be employed during case-based problem solving.

In making "similarity" operational, we must consider the different ways that similarity effects the update and retrieval prosesses of conceptual memory. During update, the judged similarity between a new concept and the concept then existing at a given node in memory must be determined. When this relationship can be determined, the new concept can be properly integrated into memory and the overall organization maintained. The required judgement is provided by a "similarity operator" that can determine the relationship between a new concept N and an old concept O such that one of the following relationships holds.*

1. N and O are the same concept (conceptual identity),
2. N is more special than O (conceptual subclass),
3. N is more general than O (conceptual superclass),
4. N and O share a common parent (conceptual siblings),
5. N and O have no relationship with each other.

We describe such a "similarity operator" in the next section. Our update process depends on this operator to identify one of the above relationships which can then be used to direct processing to the appropriate place in memory. When a new concept is being integrated into memory, we use only this local knowledge about the relationship between the new concept and an existing node in memory to direct memory traversal and update.

During retrieval, a similarity judgement is once again required to determine where a concept would be in memory if it had been "seen before." This requires performing the same comparisons as described above for memory update. The same "similarity operator" is used as part of a retrieval process that we describe later. Using these same five relationships as a guide, this retrieval process can also traverse memory to locate a desired concept when it exists in memory or, failing that, locate a concept in memory that is either a sibling or a more general concept. This is the basis for our analogical reminding.

In addition to using similarity judgements to direct the basic retrieval process, a different judgement of similarity is required when the retrieval mechanism returns more than one reminding. In this instance, we require a similarity based evaluation function that will identify the most appropriate case for focussed use in problem solving. We will describe such a similarity based evaluation function in a later section.

6.3 Organizing and relating cases in memory

In chapter one, we introduced generalized episodes as organizing structures for a dynamic memory. These structures hold generalizations, called norms, compiled from the cases they organize. Individual cases are indexed off these structures by those features that differentiate them from the norms. The concepts used for organization (i.e., as generalized episodes) are the same ones used for representing the original problem. Thus, there are generalized episodes corresponding to "physical disputes," "disputes over food," etc. When a new case is used as a memory probe, a set of generalized episodes for the case is selected. Indices already there, corresponding to the features of the case, are then traversed. In this way, the most similar previous cases are found. When adding a new case to memory, the same process is used, and in addition, indices are created for each feature of the case that differentiate it from the norms of the generalized episode and that is not already an index.

This section explains in more detail how generalized episodes were used to implement a conceptual memory organization for the MEDIATOR program. This memory model satisfies the requirements outlined in the preceeding section.

6.3.1 Types of generalized episodes

We have identified three types of generalized episodes useful for organizing cases in case-based problem solving: component, classification, and tactical ones. Component and classification generalized episodes are used during the understanding phase of problem solving, while tactical ones are used during planning.

The understanding phase of problem solving, described in chapter three, requires retrieval of previous cases based on the similarity of their components. For example, a new dispute whose disputed object is candy should elicit reminding of the case in memory with the disputed object most similar to candy. We accomplish this by organizing cases in generalized episodes associated case components. With respect to disputes, this means disputed objects, disputants, dispute arguments, etc. (as discussed in chapter two), are used as generalized episodes. Within each of these generalized episodes, indexing is by additional features of the selected component. Thus, in generalized episodes based on "disputed objects," indexing is by features of objects involved in disputes that are organized in memory. As a result of this organization, disputes about food will be organized in the same place, within the generalized episode organizing the "disputed object" concepts. By the same method, disputes whose disputants are polities will be organized together in the generalized episode used for the "disputants" component. In this way, knowing only the component features of a dispute (i.e., without knowing the dispute type) will be sufficient to allow reminding to occur. Once cases are recalled, the dispute can be further classified into "physical" "economic" or

"political" categories. The generalized episodes which allow this type of reminding are called component generalized episodes. This type of structure enables the bottom-up reminding that initiates problem solving actions such as classification. Figures 6-4 and 6-5 shown later provide sample generalization heirarchies that correspond to the knowledge used to construct specialization indices for "disputed objects" and "disputants" component generalized episodes.

Classification type generalized episodes correspond to problem types in the problem domain, in this case "physical," "economic," or "political disputes." Each problem type makes reference to appropriate generalized strategies for resolution (e.g., "divide equally" plans for physical disputes). While component generalized episodes allow the types of reminding that help in selection of a problem type, classification type generalized episodes facilitate selection of a plan for problem resolution. Indexing in classification type generalized episodes is by features of each of the components of the problems in a class. At the highest level, indexing is by problem components (e.g., disputants, disputed objects, etc.), below that indexing is the same as in component generalized episodes. Note, for example, in Figure 6-1 that indexing for "disputants" and "disputed objects" within the "physical disputes between children over food" partially mirrors that of "disputants" (Figure 6-3) and "disputed objects" (Figure 6-4). Notice that information based on the Panama Canal dispute (a political dispute) is missing in Figure 5-1, while information about the Korean War (a physical dispute) is available.

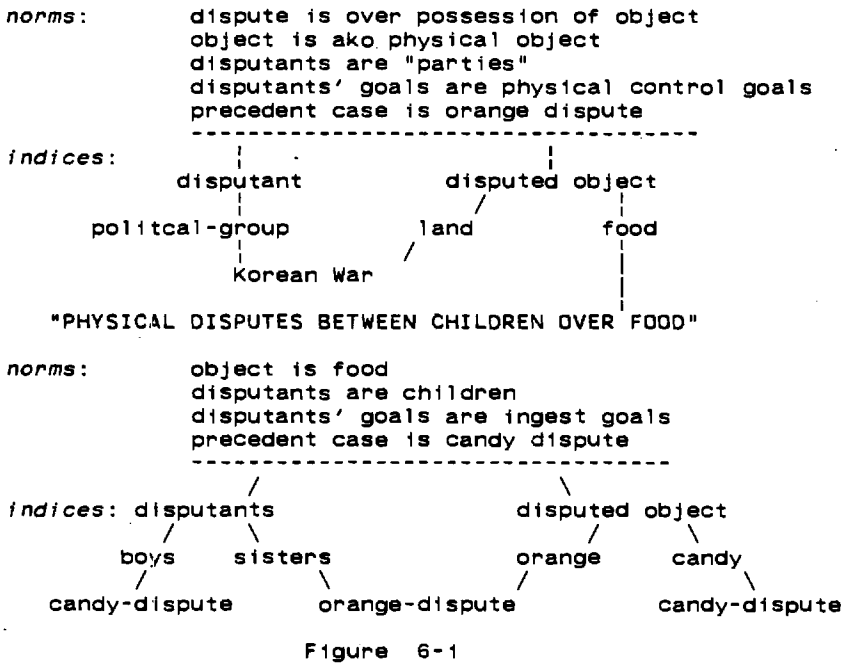


Figure 6-1

In our problem solving framework, two fundamentally different types of problems require the use of component and classification type generalized episodes during understanding: domain problems and reasoning failures. We thus have generalized episodes corresponding to the components and types of each of these different problem types. We have already mentioned the component generalized episodes for disputes. For failures, the component generalized episodes correspond to components of failures (e.g., the problem classification, the plan attempted) and the classification type generalized episode corresponding to classes of failure (e.g., "misclassification," "miselaboration;" see Figure 5-12). These allow cases to be recalled during the failure understanding phase (i.e., blame assignment). An example of this was shown in section 5.4.4. Below is an illustration of "failures due to goal misunderstanding," a failure-based classification type generalized episode.

"FAILURES DUE TO GOAL MISUNDERSTANDING" GENERALIZED EPISODE

norms: dispute ako physical dispute
object is ako physical object
disputants are "parties"
disputants' goals are physical control goals
plan attempted is "one cuts, the other chooses"
precedent case is orange dispute

indices:

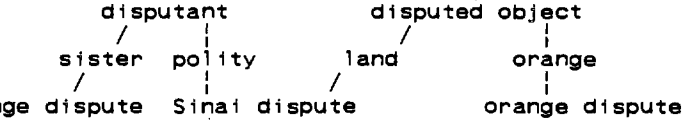


Figure 6-2

The third type of generalized episode is tactical. Tactical generalized episodes come into play during planning and correspond to experiences with a particular plan, with the components of a plan (e.g., contracts), or with a mediation type (successful or unsuccessful mediation). Those that deal with plans, for example, describe known preconditions, implementation details, and expected results of plans, and organize cases in which the plan was used. These are all important sources of knowledge necessary to support the decisions made during the planning stage of problem solving. In the mediation domain, we have tactical generalized episodes corresponding to mediation plans (e.g., "divide equally," "take turns," "divide unequally"), remediation plans, i.e., plans for recovery from particular planning failures (e.g., "change planning policy" or "infer goal from resulting events"), as well as contracts (e.g., "divided object contract"), and mediation experiences (e.g., "unsuccessful mediations"). Tactical generalized episodes are used during the planning stages, as described in chapter four, to determine whether a suggested plan is appropriate (using its known preconditions), to find a means of instantiating a plan (by looking at previous contracts), as well as to predict and evaluate the consequences of using a suggested plan. During failure recovery, tactical generalized episodes associated with remediation plans help in correcting problem misinterpretations and selecting alternative resolution plans (see section 5.5). Tactical generalized episodes also allow a problem solver to retrieve generalized experiences based on success or failure under specific conditions. Figure 6-3 provides an illustration of the "one cuts, the other chooses" tactical generalized episode with two cases organized within it.

"ONE CUTS, THE OTHER CHOOSES" TACTICAL GENERALIZED EPISODE

norms: dispute is ako physical dispute
object is ako physical object
disputants are children
disputants' goals are physical control goals
plan attempted is "one cuts, the other chooses"
preconditions are ...
precedent case is candy dispute

indices:

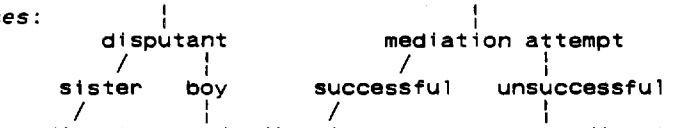


Figure 6-3

6.3.2 Organization around conceptual components

In chapter two, we identified a number of concepts that make up the MEDIATOR's domain knowledge about the mediation of disputes (e.g., disputants, disputed objects, etc.). At the same time, we specified that these mediation concepts would be represented by a set of conceptual primitives. These same concepts are used for organization of cases. This set of primitives effect a conceptual model of the MEDIATOR's knowledge of the problem domain. These primitives must be stored and related properly in the MEDIATOR's conceptual memory in order for case-based reasoning to occur.

Each conceptual component of a mediation case, such as the disputed object, is described in terms of a semantic model that organizes all concepts into a generalization hierarchy. The two most notable features of this semantic knowledge is that it provides an *instance* language and a *generalization* language (Mitchell, 1981). The instance language occupies the leaf nodes of the generalization hierarchy and is made up of those primitive concepts necessary to represent the MEDIATOR's specific case experiences. For example, candy1 is an instance of the primitive M-CANDY and orange1 is an instance of the primitive M-ORANGE which are both elements of the instance language for the MEDIATOR's disputed object knowledge.

As soon as we contemplate how these instance language concepts should be related to each other in memory, we realize the importance of the generalization language. As part of our functional requirements, we indicated that we wanted to organize these problem concepts in such a way as to (1) enhance the learning of cases involving new concepts and (2) facilitate the reminding of similar cases during problem solving. A simple minded approach to the storage of instance language primitives would be to simply link them together in a list. This would certainly provide the capability to learn a new case, but it would not provide the kind of organization that promotes analogical reminding. Nor does an unorganized list address our performance requirements that retrieval not slow down as the number of cases gets large.

The generalization language, which occupies the non-leaf nodes in the generalization hierarchy, provides the knowledge of how the instance level and other general concepts are related. This knowledge is used during memory update to direct the construction of episodic memory. During retrieval, this knowledge is used to select indices that allow concepts to traverse the specialization links in episodic memory to the most specific level possible. These aspects will be demonstrated in later sections on update and retrieval.

There is a generalization hierarchy for each important component of the domain. This knowledge provides the explicit semantic model of the problem domain. For example, disputed objects as shown in Figure 6-4 are, in general, physical objects. Given three disputed objects a candy bar, an orange, and an avocado, we use the generalization primitives M-FRUIT, M-FOOD, M-CONSUMABLE-OBJ, M-FUNCTIONAL-OBJ, M-SPLITTABLE-OBJ, and M-PHYS-OBJ (see section 2.3.5) to construct a generalization hierarchy for "disputed objects."

A PARTIAL GENERALIZATION HIERARCHY FOR DISPUTED OBJECTS

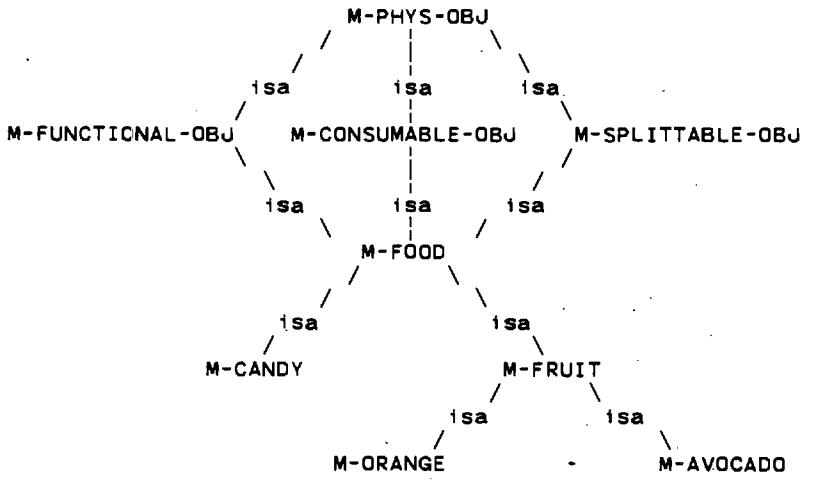


Figure 6-4

We can do the same for "disputants," who in general, are higher animates. We call them "parties."

A PARTIAL GENERALIZATION HIERARCHY FOR DISPUTANTS

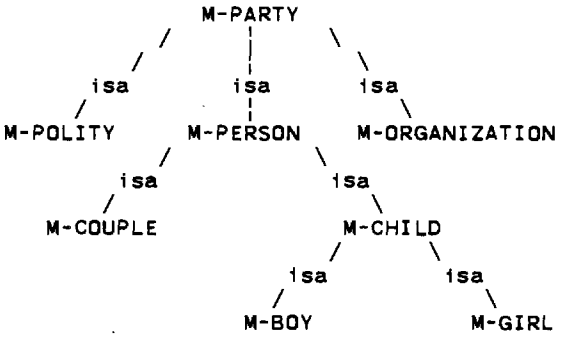


Figure 6-5

As illustrated for the "disputed object" component shown in Figure 6-4, we can construct a generalization language so that it provides a semantic model of each important case component. As a problem solver resolves cases, knowledge such as this allows the experiences associated with the instances of these concepts to be related in episodic memory. The semantic model, in effect, acts as a "blueprint" that guides the construction of episodic memory during the update process. Thus when trying to relate the cases which contain the concepts M-ORANGE and M-AVOCADO within the mediator's experience with disputed objects, the concept M-FRUIT provides the conceptual link between them. This semantic link becomes a generalized episode when the cases collide in episodic memory and the generalization is made. It is in this way that we say episodic memory evolves according to the semantic model. The generalization process that infers an M-FRUIT generalized episode from the instances of M-ORANGE and M-AVOCADO is part of the memory update process described later.

Several features of our generalization hierarchies are notable. First, there is cross-classification (e.g., M-FOOD is classified as a merger of the concepts M-CONSUMABLE-OBJ, M-SPLITTABLE-OBJ, and M-FUNCTIONAL-OBJ). This permits domain concepts to be defined in terms of independent salient features that are important in discriminating domain concepts. For example, the MEDIATOR must know that an orange is splittable in order to consider mediation plans like "one cuts, the other chooses." Because these salient features are indepentently specified, a new instance language concept (once defined in terms of these general semantic concepts) can be integrated into the generalization hierarchy even though it was unknown up to that point. Second, there is greater depth in the overall hierarchy. This allows a greater range of differentiation among semantic concepts (and ultimately among generalized episodes once instantiated) than would be possible with more shallow taxonomies (e.g., Rifkin, 1984). Lastly, the generalization hierarchy provides a structure with the characteristic that similar concepts are closely located in semantic memory. This characteristic carries over to episodic memory since a problem solver's memory for cases evolves according to the semantic "blueprint." This matches our intuition that experiences with an orange and an avocado should be conceptually near each other.

6.3.3 Implementing generalized episodes

Generalized episodes are implemented in the MEDIATOR program as generic frames of type M-MEMORY. They provide the organizational glue used to build up case memories of related experiences. In the sense that they contain knowledge about cases, generalized episodes may be thought of as meta-knowledge structures in our implementation. Using a frame representation, each generalized episode frame contains both declarative and procedural knowledge. For example, each memory frame has a slot to identify the "type" of concept (e.g., "orange1" is of type M-ORANGE) organized within the frame. We allow a generalized episode to organize only one "type" of knowledge. This is no real limitation since larger concepts (e.g., "disputes") are one "type" of concept composed of other "types" of concepts. So within generalized episodes, we have other generalized episodes. In addition, we take advantage of the fact that procedures can be attached to frames to organize specific memory related functional knowledge. For example, we have a procedure called "reminded-of" which is attached to each memory frame instance. This procedure responds to a reminding cue by retrieving the most specific concept organized within that memory frame that "matches" the cue. Match, in this situation, means that the concept satisfies either the conceptual identity, sibling, or generalization relationships in that respective order of preference. Our implementation of generalized episodes is reflected in Figure 6-6 below. In Figure 6-6, we show a M-MEMORY frame with its associated slots for attached procedures. The usual fillers for the slots or the function performed by the procedure is also briefly described opposite the slot.

FRAME USED TO IMPLEMENT GENERALIZED EPISODES

M-MEMORY
- *type-name*: a symbol representing some mediation concept
- *norm*: an instance of the type-name whose slot fillers are either specific instances of the appropriate type or another m-memory frame representing another generalized episode
- *event*: the precedent case associated with this frame
- *specializations*: property list of specialized type-names with their associated m-memory frames.
- *reminded-of*: retrieval procedure that returns the most specific norm from this frame for a given cue.
- *recall-containing-event*: retrieval procedure that returns the event filler from the frame that best matches a given retrieval cue.
- *reconstruct*: a procedure that constructs a prototype from the norm slot of this frame.
- *reconstruct-special*: another prototype constructing procedure which constructs its instance according to the best match to a given model.

Figure 6-6

Because generalized episode frames are generic and can organize knowledge about any concept, the "type-name" slot is necessary to indicate the type of knowledge organized within a particular frame instance. The "norm" slot serves two purposes in this implementation. First, those features common to all the experiences organized within this frame are indicated by instance values filling the appropriate slots of the norm instance. Thus if all the experiences have the same fillers for each of the slots of this concept (e.g., the cases differ on some other component feature), then the norm would be a fully specified instance of the "type-name" concept. This means that the norm slot, in this case, provides an operational representation of a prototype (i.e., a fully instantiated exemplar of the mediation concept). Second, since it is unlikely that each case will have exactly the same value for all the slots in the norm (this corresponds to those situations where there is already a case indexed by a feature), then a new M-MEMORY frame (generalized episode) is created to organize the differences within this slot of the norm. In this instance, the concept from the generalization language that links the two instances will be represented in the prototype (e.g., fruit represents the generalization of orange and avocado in the "disputed object" slot of a generalized dispute frame). The result is that the "norm" slot provides the capability of both constructing a prototype of the concept represented by the frame (using one of the reconstruction procedures), and also locating specific cases that differ according to those features identified within the norm. This approach implements the within concept organization for each specific problem component (e.g., disputes, disputed objects, disputants, etc.).

The "event" slot in the M-MEMORY frame implements the notion of the precedent case. The case filling the "event" slot is immediately retrievable as an exemplar case which contains the mediation concept represented by this frame. This allows a similar case to be returned even when memory traversal terminates at a level whose concept is more general than the retrieval cue.

The "specialization" slot for a M-MEMORY frame links the concept organized within this frame to other M-MEMORY frames organizing more specific concepts. These frames are reachable from the current concept by explicitly identifying the "type-name" of the more specific concept. This is the sense in which we require memory search to be directed. As a problem solver accumulates case episodes, conceptual memory will evolve according to the generalization hierarchy for each problem component. For example, the MEDIATOR's knowledge of "disputed objects" evolves according to what it knows about how "physical objects" are related. This knowledge comes from the generalization language, part of which was illustrated in Figure 6-4 for "physical objects."

To further illustrate our methodology, the frame representing a generalized episode for the disputed object concept of "fruit" after being created as a generalization from Orange-Dispute-0 and Avocado-Dispute-0 is shown in Figure 6-7 below. Notice that Orange-Dispute-0 is retrievable either as the precedent case via the "event" slot or via the "orange" specialization of "fruit." In the latter case, retrieval would require traversal of the M-ORANGE link to the M-MEMORY frame organizing cases which involved "oranges."

FRAME ORGANIZING EPISODES WHERE DISPUTED OBJECT WAS "FRUIT"

```
M-MEMORY; i.e., dispute cases where the disputed object was a
                type of fruit
    type-name: M-FRUIT
    norm: M-FRUIT ; i.e., an instance of type M-FRUIT
              isa: M-FOOD
                       isa: M-SPLITTABLE-OBJ
                                is-splittable: t
                                has-as-parts: (M-SEED M-PEEL
                                               M-PULP M-JUICE)
                                is-part-of: nil
                       isa: M-CONSUMABLE-OBJ
                                is-consumable: t
                       isa: M-FUNCTIONAL-OBJ
                                normal-usage: M-INGEST
    event: orange-dispute-0
    specializations: (M-ORANGE              M-AVOCADO)
                              |                    |
                  orange-dispute-0       avocado-dispute
```

Figure 6-7

6.3.4 Organizing different generalized episodes

At the top level in episodic memory, we provide a structure that is used to organize the many different component, classification, and tactical generalized episodes that are needed for case-based reasoning. From this level, each generalized episode can be probed by retrieval cues to produce the case containing the concept most similar to the cue. Component generalized episodes are labeled to indicate the concept types around which cases are organized (according to the semantic "blueprint"). For example, cases organized with respect to the disputed object component of disputes are accessible from a top level generalized episode called "memory-for-objects." Another generalized episode called "memory-for-goals" organizes cases with respect to disputant goals. At the top level, all such generalized episodes are packaged into a single globally accessible frame of type "M-LTM" as shown in Figure 6-8. This frame provides the explicit organization between different generalized episodes in the MEDIATOR's long term memory. "Physical disputes, for example, are contained within the MEDIATOR's "memory-for-disputes" since they are a specialization of "disputes."

MEDIATOR'S LONG TERM MEMORY FRAME

M-LTM
- *memory-for-objects*: a component M-MEMORY frame
- *memory-for-parties*: a component M-MEMORY frame
- *memory-for-goals*: a component M-MEMORY frame
- *memory-for-disputes*: a classification M-MEMORY frame
- *memory-for-mediation-plans*: a tactical M-MEMORY frame
- *memory-for-mediation-experiences*: a tactical M-MEMORY frame
- *memory-for-arguments*: a component M-MEMORY frame
- *memory-for-contracts*: a tactical M-MEMORY frame
- *memory-for-failures*: a classifcation M-MEMORY frame
- *memory-for-remediations*: a tactical M-MEMORY frame
- *update-buffer*: workspace used during memory update

Figure 6-8

As a top level node in a problem solver's long term memory, this frame represents the entry to episodic knowledge and provides the structure that organizes the many different individual component memories. Retrieval of previous cases begins by accessing this global frame and using the appropriate components of the current case as retrieval cues for each of their corresponding generalized episodes. Our architecture includes this long term memory frame as one component of a globally accessible blackboard (Erman, et al., 1980; Hayes-Roth and Hayes-Roth, 1979). The overall structure of the blackboard is discussed further in section 5.6.2.

6.4 The update process

Memory update is a two step process. First, the relevant features of the new case are determined via the process known as "index selection" (Kolodner, 1984). Second, the case is indexed within the appropriate generalized episode as determined by the state of the existing "norms" and the similarity relationship that exists between the concepts organized there and the concepts within the new case. The new case is indexed within a generalized episode, after it has been "pushed" down the memory hierarchy to its most specific level. When the new case is unique within a generalized episode according to the concept organized there, a new index is created and the case is indexed there. When there are one or more other cases indexed according to the same differences, then a new or updated generalized episode is created.

When a case is being processed for inclusion in a conceptual memory, it needs to be indexed within all component and tactical generalized episodes associated with the case. Thus, for example, we need to insure that our conceptual "memory-for-disputants" indexes the case via the features of the disputants, our "memory-for-objects" indexes the case according to the features of the disputed object, etc. For each important component of the case, regardless of the generalized episode, the effect of a new concept on a specific generalized episode depends on which of the following five similarity relationships exist between the two concepts:

FIVE SIMILARITY RELATIONSHIPS THAT MAY EXIST BETWEEN TWO CONCEPTS

1. the new concept *is more general than* the old concept.
2. the new concept *is more specific than* the old concept.
3. the new concept *is the same as* the old concept.
4. the new concept *has a parent concept in common* with the old concept.
5. the new concept *has no relationship* to the old concept.

Figure 6-9

In order to determine which of these specific relationships exist between two concepts, a similarity operator is used. This operator enables a problem solver to make the necessary similarity judgements and based on the established relationship between two concepts indices can be selected and memory traversal directed.*

---

*Mitchell (1981) describes the importance of a partial ordering of concepts based on the "more-specific-than" relation. This relation, of course, corresponds to the second relationship listed above. As we discussed, there are other relationships that are possible and we need to account for them in a fuller conceptual memory model such as ours.)

6.4.1 Index selection

The first step in updating or retrieving a case from a generalized episode is the selection of the appropriate indices so that the traveral process can find the appropriate place for the case. Traversal, whether for update or retrieval, requires following the appropriate indices down the memory hierarchy until the correct location is found. This means that any case other than the one accessible from the top level cannot be retrieved unless the appropriate indices can be specified at each intermediate point in memory.

One rather obvious way to ensure that a case is pushed down as far as possible in the specialization heirarchy is to let the traversal process enumerate all the indices emanating from a generalized episode to insure that no specializations are overlooked. If we allowed this, then traversal time would grow in proportion to the number of features indexed in memory. To see this, imagine a retrieval process that has to enumerate hundreds or thousands of indices at each of several levels in order to traverse memory. By restricting retrieval and update traversal to a directed search, we more closely immitate the near constant retrieval time of people (Smith, et al., 1978). One of our performance requirements, you may recall, was a negligible slowdown in retrieval time as more cases are added to memory. As a consequence we face the problem of needing a mechanism to specify indices for directing traversal.

This problem is addressed by using the knowledge provided by the semantic model of each component and a similarity judgement that can differentiate the five similarity relationships shown in Figure 6-9. This mechanism allows indices to be selected for memory traversal during retrieval in a fashion analogous to that used when the memory update process is traversing memory to locate the proper spot for adding a new case. To illustrate in somewhat more detail how similarity is determined, consider what this means for the disputed object candy1. Candy1 is an instance of the mediation primitive M-CANDY. In turn, M-CANDY is a specialization of M-FOOD, as was shown in Figure 6-4. What does this mean to candy1? If explicitly queried, candy1, or any other instance of a conceptual primitive, can verify whether or not it is an instance of a given concept. Not only can each primitive verify its dependency, but it can also explicitly produce it when requested. We believe this capability is essential in order that index selection and memory traveral be a directed process. The examples below might make this clearer:*

(1) candy1 isa M-CANDY? ==> true
(2) candy1 isa M-FOOD? ==> true
(3) candy1 isa M-PHYS-OBJ? ==> true
(4) candy1 isa M-DISPUTE? ==> false
(5) candy1 depends-on ? ==> (M-CANDY M-FOOD ... M-PHYS-OBJ)

In terms of locating the appropriate level in a conceptual hierarchy, this similarity judgement allows the index selection process to (1) quickly verify that a conceptual instance (if it exists in the generalized episode) is located on or below the current level and (2) select the appropriate index to direct traversal. This entire process must be efficient since it will be performed many times during update and retrieval. One way to improve its efficiency is to verify that a probe is "on the right track" in order to prevent unnecessary search. The index selection process accomplishes this by simply querying the cue upon entry to a generalized episode as shown in example (3) or (4) above.

Using (3) as an example of entry to "memory-for-objects," if the top level generalized episode is of type M-PHYS-OBJ then via (3) we verify that the concept M-CANOY, if it has been a component of a previous case, would reasonably be located within those cases organized below the M-PHYS-OBJ primitive. Using (4) let's assume that candy1 is used as a cue for "memory-for-disputes." In this case, the query will quickly verify that candy1 is not of the appropriate type for this generalized episode, thus avoiding fruitless search.

To select the correct index for traversal, notice that the cue concept can be queried to specify, in order from most to least specific, those primitive concepts on which it depends. This effectively presents an ordered list of plausible specialization indices, as illustrated by example (5) above. Thus in this case, candy1 would first probe for a specialization index with a value of M-CANOY, followed by a probe for an index of value M-FOOD, etc. This also corresponds in part with what Kolodner (1984) calls "index fitting." Thus, traversal is directed not by any implicit knowledge of the contents of memory (i.e., what indices are available), but only by explicit knowledge available locally from the cue itself.

---

*This is implemented on the LISP Machine using the "typep" and "flavor-depends-on-all" functions. See Weinreb and Moon (1981) for details.

6.4.2 *Adding a new case to memory*

Guided by the knowledge provided by the index selection process, the traversal proceedure locates the appropriate place in the conceptual memory. The actual processing that is performed next depends on what cases are already indexed in memory at that location and their similarity to the new case. The memory update processing is different according to which of the five similarity relationships presented in Figure 6-9 apply.

If the new concept is more general than the current concept organized at this memory node, then the update process needs to create a new generalized episode frame to organize this concept and insert it into the memory tree above the current node. This, of course, requires that all generalized episodes linked above the current node be reconnected to the new concept node and the current node indexed below as a specialization. For example, suppose that the first case integrated into "memory-for-objects" involved a candy bar, but the second involved "food". Because the concept "food" is more general than "candy bar," it needs to be inserted above "candy bar" in conceptual memory. The norm for food in this situation would be a candy bar and the precedent case would be Candy-Dispute-0.

If the new concept is a specialization of the current concept, then three further actions are possible. First, if there is already a generalized episode indexed by this concept then traverse its link and treat it as specified for case (3), discussed below. Second, if there is an intermediate concept (e.g., food is intermediate between candy and physical object) indexed from the current node, then traverse that link. It will be treated as a specialization of the concept at that node as well. Third, if neither of the first two options apply, then index the new concept below the current node.

The third situation listed in Figure 6-9 was concerned with the condition when the new concept was identical to the concept organized at the current memory node. In this instance, the norms of the concept need to be updated. Each corresponding feature of the two instances of the same concept are compared, if they are both equivalent then no change is made. When the norm has a generalized episode filling the corresponding slot, the features from the new concept are used to recursively apply the update algorithm to this generalized episode. When the feature values are not equivalent, then a new generalized episode is created for the slot and the two different feature values are indexed below. To illustrate this situation, imagine a second candy bar case. Unless there is something unique or distinguished about the feature values of this second candy bar, then no change will be made to the conceptual memory. But, if the second candy bar were spoiled (e.g., the "is-consumable" slot is nil) then this difference will require that the two cases be indexed according to this feature.

The fourth situation is when the new concept is a sibling of the concept at the current node. This case arises the first time two instance language concepts must be related to each other. For example, "candy" and "orange" are sibling concepts that fail to satisfy either of the similarity relationships (1), (2), or (3) described above. So after the first primitive concept has been indexed into memory, the next will need to somehow be integrated relative to the other. This is the classic case of generalization, and requires that the concept food be inferred from the two concepts candy bar and orange. This is accomplished by an intersection of the two lists of parent concepts to locate the "most specific common parent" of the two concepts. For our simple candy and orange example, we know from the generalization language for physical objects that "candy" is dependent on the concepts: M-CANDY M-FOOD ... M-PHYS-OBJ. The "orange" concept is dependent on the concepts: M-ORANGE M-FRUIT M-FOOD ... M-PHYS-OBJ. With this knowledge, locating "food" as a generalization of the "candy" and "orange" concepts is reduced to locating the common concept in the generalization language.

Even though this method of inferring generalizations is limited to the generalization language used to model the domain, it has several advantages. First, it produces no "bad" generalizations, assuming the domain model is correct. Two concepts are sufficient for generalization to occur. Other approaches need some number of cases before allowing generalization and require recovery procedures for repairing the effects of bad generalizations (Kolodner, 1984). Depending on the conceptual model, the above method of generalization allows inductive leaps beyond the information that can be obtained strictly from a narrow view of the concept itself (Mitchell, 1981). Because each concept "knows" the generalization primitives that it depends on, generalization can occur with much more assurance. We, therefore, make no provision for recovery from bad generalization.

The final situation, number (5) in Figure 6-9, that can exist between a new concept and the concept at a memory node is when the two concepts are completely unrelated according to the conceptual domain model. This situation is inferred when all of the previous relationships fail to apply. If the new concept has no relation to the current node, then it probably means that an attempt has been made to update a generalized episode with the incorrect type of information (recall that we restrict each generalized episode instance to organizing cases with respect to a single concept). In this case, no update takes place and an update error is indicated. Detecting these kinds of errors helps maintain the integrity and consistency of the generalized episodes by trapping updates that violate the domain model. The update error could be caused by a simple miscoding, or could indicate an area where the model possibly needs modification. We have not addressed these issues.

6.5 Reminding

Based on the description of generalized episodes provided in chapter one and elaborated in the preceeding sections, reminding happens in the following way: During understanding, component generalized episodes associated with individual component features of the problem are traversed to the lowest level possible. This allows reminding of cases with similar component features. These cases then suggest possible classifications for the problem. Traversal continues in identified classification generalized episodes, allowing a problem solver to recall the most specific case in memory that is similar (according to our similarity relationships) to the case being processed. Final classification and the availability of previous similar cases classified the same way allows suggestion of plans for resolution. Each plan corresponds to a tactical generalized episode, which is traversed using the features of the dispute in order to discover the best way to apply the plan. Memory search, during failure recovery, happens the same way, this time using generalized episodes corresponding to classes of failures and tactical generalized episodes corresponding to remedies.

Note that the memory traversal process results in the retrieval of previous cases most similar to the current case. In effect these are "near-misses." We never expect to have exact matches. his retrieval process is summarized in Figure 6-10, which explains how reminding happens in the MEDIATOR.

THE MEDIATOR'S ALGORITHM FOR RETRIEVING CASES FROM MEMORY

1. For each component of the given problem representation, probe in parallel the generalized episode corresponding to the knowledge organized for that component. The features of the problem component act as a retrieval cue.

2. Descend the generalization heirarchy for each chosen generalized episode to the most specific level reachable for the given cue. This is done by choosing the first specialization index at each level that allows traversal to the next lower level.

3. If traversal terminates at a memory node whose concept is more general than the cue or if the traversal terminates at a memory node with a fully instantiated norm (i.e., it is not another generalized episode), then return the precedent case from the "event" slot of the memory frame at that point.

4. Else recursively apply this algorithm for each component of the norm which has a generalized episode for a filler, using the corresponding components of the old cue as the new cues for this set of new probes.

Figure 6-10

This retrieval process has some notable characteristics. First, as long as there is at least one case in memory, this process is guaranteed to retrieve the case that is the best "near-miss" according to the conceptual model of each mediation component described by the generalization language. Second, there is no guarantee that the case retrieved will be a useful case for analogical reasoning. Thus a post-retrieval process is necessary to choose the most appropriate case from all those retrieved. Third, there is no elaboration during retrieval; although the recursion in step four can be viewed as a kind of elaboration. When traversal is blocked, the precedent case is returned as the "best-fit". Fourth, there are many opportunities for retrieval to occur in parallel when efficiency is a concern.

6.5.1 A retrieval example

To illustrate the reminding process in more detail, we will present a simple example that demonstrates the memory structures before retrieval and the traversal paths that allow cases to be found. The example is taken from the Sinai dispute case, which was used to demonstrate the behavior of the MEDIATOR program at several points in our discussion.

Four cases have been processed into conceptual memory prior to beginning the Sinai dispute. The cases and their order of processing are as follows: the Korean conflict, the Panama Canal dispute, Candy-Dispute-0, and Orange-Dispute-0.* A simplified schematic of conceptual memory prior to the Sinai dispute is shown in Figure 6-11.

---

*These cases have all been presented earlier. The complete collection of our cases can be found in Appendix A.

MEDIATOR'S CONCEPTUAL MEMORY PRIOR TO THE SINAI DISPUTE

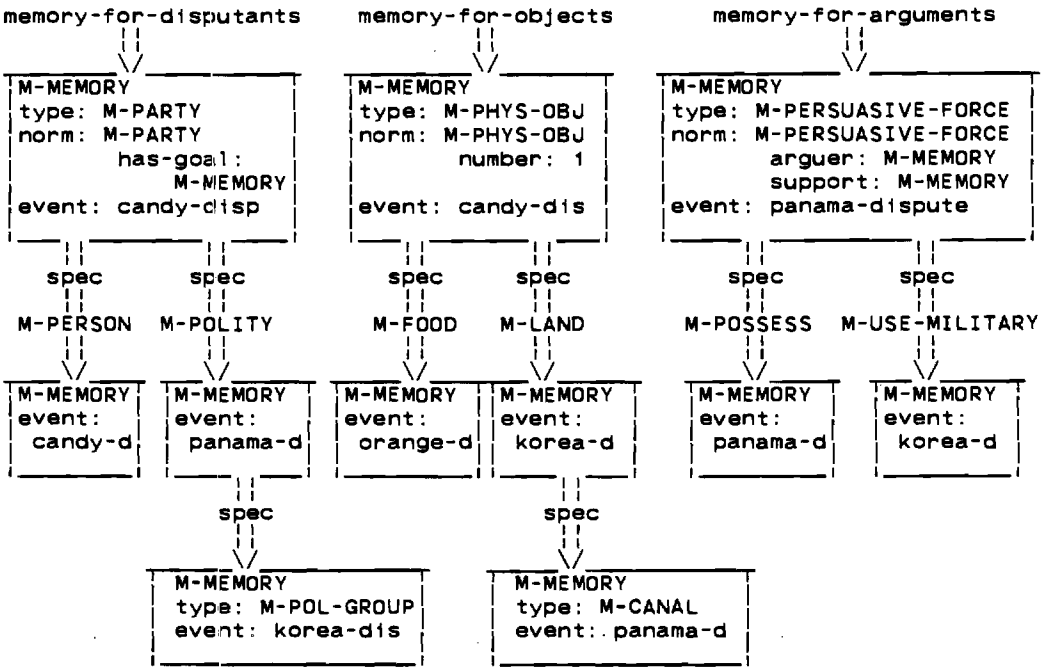


Figure 6-11

This schematic shows a simplified view of the three generalized episodes corresponding to the mediation components for disputants, disputed objects, and arguments. At the top level, the generalized memory structures reflect the kinds of generalizations one might expect. For example, the generalization of M-PERSON and M-POLITY yields M-PARTY (i.e., disputants are parties to a dispute). The "norm" slot is filled by an instance of the concept labeled by the "type-name" slot. Note that the generalized M-PARTY contains a generalized goal organized within the generalized episode indexed by the "has-goal" slot. As a result of processing the four previous cases, there are six generalized episodes indexed below the three top level frames via specialization relations (i.e., "spec" in Figure 6-11). Some of these in turn have other generalized episodes indexed below them as further specializations (e.g., M-POL-GROUP is a primitive used to represent a coalition of polities, in this case the UN forces in the Korean Conflict). Notice that at each level the precedent case is immediately available via the "event" slot.

With the retrieval cues provided by the input representation of the Sinai dispute, the parallel paths through the conceptual memory schematic are highlighted in Figure 6-12 below:

RETRIEVAL CUED BY THE SINAI DISPUTE

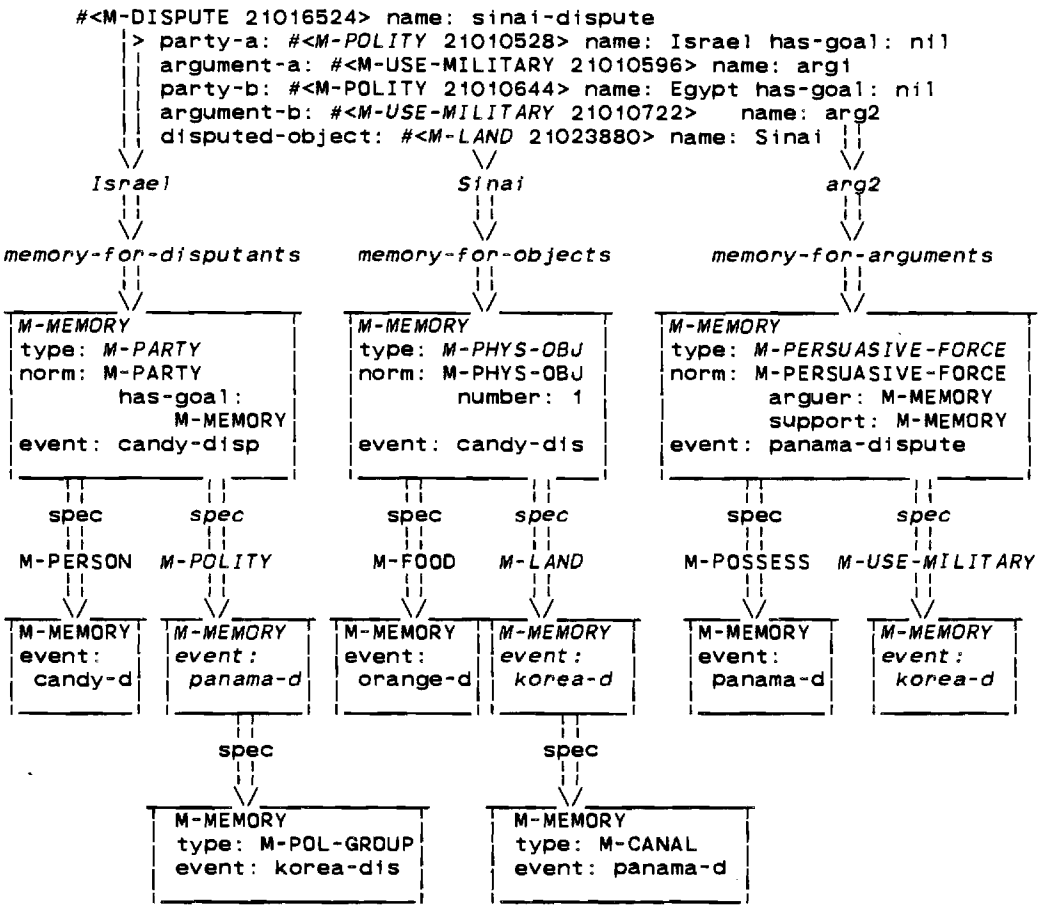


Figure 6-12

The retrieval illustrated in Figure 6-12 is cued by three components of the input dispute representation: the disputant filling the "party-a" slot, the argument filling the "argument-b" slot, and the object filling the "disputed-object" slot. These three retrievals all go on in parallel, however, we will address only the retrieval cued by the disputant. First, the filler for the "party-a" slot which contains a representation for Israel, as an instance of type M-POLITY, is used to probe the "memory-for-disputants" generalized episode. In Figure 6-12, the top level of the "memory-for-disputants generalized episode is a memory node for the concept M-PARTY. Two tests are conducted at this level to determine the similarity relationship between the probe concept, M-POLITY, and the M-PARTY concepts. These tests are shown below.

```
#<M-POLITY 21010528> isa M-PARTY?  ==> T
M-POLITY equal M-PARTY?  ==> nil
```

The first test confirms that if a similar event had occurred it would be located at or below this level in memory. The second test indicates that the cue concept and the concept at this level are not conceptually identical. Next the cue is queried for its dependency in order to probe for specialization links that can be traversed to a lower level. Since the primitive M-POLITY depends on M-POLITY and M-PARTY, traversal of the M-POLITY specialization link then allows the retrieval process to move to the next lower generalized episode. The same two similarity tests administered at this level show that the cue concept and the concept located there in memory are identical. The "event" slot provides the precedent case, the Panama Canal dispute, that is returned as a reminding. This process is repeated for the other generalized episodes associated with other dispute components. This retrieval process is what underlies the external behavior exhibited by the MEDIATOR in Figure 6-13:

MEDIATOR'S I/O BEHAVIOR DURING RETRIEVAL CUED BY THE SINAI DISPUTE

```
(mediator sinai-dispute t)
Considering the following problem:
   Israel and Egypt both want the Sinai,
   which has been presented as ako M-DISPUTE.
ATTEMPTING TO RECALL SIMILAR PROBLEMS IN ORDER TO CLASSIFY THIS ONE...
  looking for disputes with similar disputants or with similar goals...
  looking for disputes whose disputants used similar arguments.
  looking for disputes involving similar objects...
   reminded of the US and Panama are quarreling over the Canal
    because both disputants were of type M-POLITY.
   reminded of Korea was in dispute
    because the object in that case, KOREA,
     was the same type object (M-LAND) as SINAI and
     because the argument used in that case, the military force argument,
                                was of the same type, M-USE-MILITARY, as this dispute.
There were two previous cases found.
   #<M-POL-DISPUTE 21016670> was the case where the US and Panama
      are quarreling over the Canal.
   #<M-PHYS-DISPUTE 21016524> was the case where Korea was in dispute.
```

Figure 6-13

6.5.2 *Schank's classes of reminding*

Schank (1982) has defined five broad classes of reminding, which are repeated below:

1. Physical objects can remind you of other physical objects.
2. Physical objects can remind you of events.
3. Events can remind you of physical objects.
4. Events can remind you of events in the same domain.
5. Events can remind you of events in different domains.

The memory organization and retrieval processes described above exhibit each of these classes of reminding. To demonstrate the first class, notice that the Sinai in Figure 6-11 and 6-12 caused the reminding of Korea. The generalized episode that we call memory-for-objects is specifically designed to support class-one and two remindings. Once Korea is recalled its associated case is available, thus demonstrating a class-two reminding. Class-three remindings are new cases that prompt remindings in all of the appropriate components of conceptual memory. Thus class-three remindings subsume both class-one and two in this view. For example in the above case, the Sinai, the physical object part of the dispute "event," caused the reminding of the physical object Korea; which in turn caused the reminding of the Korean conflict "event." Remindings of type four and five were illustrated in chapter one, when the Sinai dispute led to the reminding of the Korean conflict, as already discussed. This illustrates a class-four reminding. Also in our earlier example, this reminding ultimately led to an error, the resolution of which was facilitated by the reminding of the orange dispute. This, albeit indirectly, is a reminding of the fifth-class, since a failure in an international mediation attempt caused a reminding in an interpersonal mediation.

6.6 *Selecting the most applicable case from memory*

Our organization and indexing approach to conceptual memory is designed to ensure that a case which has been judged most similar to the current case is returned from each generalized episode probed. This approach is intended to increase our opportunity to be reminded of applicable previous cases. The effect of this strategy is that we are also faced with a requirement to identify and select the most applicable case from all those identified as being similar to the new case. This section describes the methodology used to satisfy this requirement. In presenting this methodology, we will first overview our technique, discuss specific issues, and then present the MEDIATOR's implementation of the general algorithm.

The screening of potential cases actually begins during the retrieval process itself. Conceptual memory is designed so that each important problem component is associated with a separate memory structure. This combined with a retrieval process which returns only one case from each memory structure effectively limits all possible remindings to one case per memory structure. For our current implementation, this means remindings are potentially limited to cases most similar to the current dispute case in terms of its disputants, disputed object, disputants' goals, and arguments. Even so, we still need a way of choosing the most relevant cases from this reduced set of remindings.

There are at least two approaches to this selection. In the first method, an "a priori" evaluation procedure, i.e., one that takes only closeness of fit to the current case into account is used to choose the best case from the set of remindings. Using this method, if selection later proves to be inapplicable (e.g., due to incompatible preconditions for the suggested plan), a second choice can be made by the same evaluation procedure. This method is acceptable when response time is an important design goal or if failures by a problem solver are not expensive or irrecoverable. "A priori" evaluation, however, may not always be reliable especially when the enviroment changes or new conditions need to be considered. It is for this reason that another method of choice must be used when more carefully investigated solutions are necessary (e.g., medical diagnosis). In this situation, an evaluation procedure is again used to rank cases, but this time a set of highly-ranked cases is identified. Suggestions from each of these highly-rated cases are then used during problem resolution, the generated possibilities are evaluated and the best one chosen.

Since the mediation domain is one where failure has relatively little risk, we use the first method. The best recalled case is chosen by an evaluation function which uses a series of elimination tests followed by a ranking of the remaining candidates based on a static priority attached to different component features. The resulting alternative cases are then sorted to produce the highest rated case. The MEDIATOR's evaluation function in the abstract is outlined in Figure 6-14:

HIGH LEVEL DESCRIPTION OF MEDIATOR'S EVALUATION FUNCTION

1. Eliminate all cases with non-matching goal relationships.

2. Eliminate all cases with non-matching goal derivations.

3. Rate and order the better cases according to the three invariance features: arguments, disputed objects, and disputants;

4. If two or more cases are equally top rated, sort the equally rated cases such that the physical dispute cases are placed ahead of economic disputes, which are placed ahead of political disputes.

5. Return the first case from this sorted evaluated list as the most appropriate (i.e., most analogous) case.

Figure 6-14

We will be explaining how the various steps of this evaluation function operate in subsequent sections. Before that, we will look at an example of the behavior of the evaluation function.

The true worth of any evaluation function is in its ability to separate superficially similar cases from those that are potentially applicable as analogies to the current case. We will illustrate how the MEDIATOR's evaluation function works to screen out superficial remindings by considering the example of the Antarctic dispute, which is repeated again below:

ANTARCTIC DISPUTE

> Fourteen nations are meeting to decide the fate of Antarctica's natural resources. One coalition is interested in developing Antarctica's resources as a means of providing income for poorer nations. Another coalition wants Antarctica protected as a natural wilderness and object of scientific investigation.

Suppose a problem solver were reminded of three cases:

(1) the Korean conflict where two international coalitions were fighting over exclusive control of Korea;

(2) the case where third-world and industrial countries both want rights to the minerals in the world's sea beds -- the third-world coalition wants to protect their future rights to these nonrenewable resources, while the industrial coalition wants to develop these resources now;

(3) the case where Israel and Egypt are fighting over control of the Sinai.

On the surface, it would appear that reminding number (2) is the most similar reminding to the current case, since that case involved coalitions with conflicting mining goals, which seems to match the Antarctic dispute. First, we can eliminate reminding number (3) using rule 1 in Figure 6-11, because it involves a concordant goal relationship rather than a competitive one

like the current case. Rule 2 eliminates (2) since the "goal relationship derivation" of the current case is different. In (2), the goal is derived from the disputants' desire to control and use a consumable (i.e., not renewable) resource, while in the current case, the goal derives from the fact that the disputants' goals require mutually exclusive uses for the disputed object (i.e., the Antarctica cannot both be preserved and developed at the same time). (1) is chosen as most applicable because its goal relationship is derived from the same mutually exclusive relationship. Thus, an analogy based on the superficial similarities between the current case and the other remindings is avoided. We will explain our evaluation function and provide sample output from the MEDIATOR program that illustrates the above case in a later subsection.

6.6.1 Some backgound on our approach

The use of an evaluation function to estimate the "value" of several different choices is well known in AI (e.g., Barr and Feigenbaum, 1981; Samuel, 1963; Simon, 1979). The technique is known in the abstract to always result in the "shortest" or least "cost" solution to a problem, given perfect knowledge about the relationships used in the evaluation function and the problem domain (Hart, et al., 1968). The problem with evaluation functions, besides the fact that we rarely have perfect knowledge, is not in their use, but in discovering good evaluation functions to use. For example, there is no theory about evaluation functions that explains why we should use goal relationships as one factor in our evaluation function. We will, however, explain why we feel it is appropriate in our domain. This may help others in developing similar evaluation functions.

Another problem with evaluation functions as used by most AI systems (e.g., Samuel, 1963) is that they are almost always additive combinations of the component feature values used to describe the problem domain.* Using additive evaluation functions leads to situations where a problem solver might decide, for example, that a mannequin was a human being because it is very similar along every dimension of evaluation except "animate." This points out the fact that in some judgements, not all component features are equally important. In the case of deciding if X is a "human being," an evaluation function needs to eliminate all candidates not having this "animate" property. It is for this reason that we use elimination tests in our evaluation function.

In previous AI research, there are two notable uses of evaluation functions that include exclusion tests as we have done. Evans (1968) used a topological-metric evaluation function to determine the similarity ratings among geometric figures. The highest rated figure was selected as the "most analogous" figure to the test figure. Evans' algorithm is outlined below:

EVANS (1968) TOPOLOGICAL-METRIC EVALUATION FUNCTION

1. Test for similarity on certain exclusionary features, eliminate figures which fail any of the following tests:

   a. both figures must be either closed curves or not
   b. the number of vertices must be the same in the two figures
   c. the number of vertices of each degree must agree with the potential analogy

2. More detailed ranking of candidate analogies are conducted on those that pass the exclusion tests. For example, each vertex is matched with a corresponding vertex of equal degree and this match is propagated to neighbor vertices until a violation is detected. All such matches are tried to determine the best fit which yields a "figure of merit" for each candidate figure. The candidate rated the highest is selected as the best analogy.

Figure 6-15

---

*This is also true for many mathematical theories of classification in psychology. See, for example, those referred to as independent cue models: Franks and Bransford (1971), Hayes-Roth and Hayes-Roth (1977), and Reed (1972). Medin and Schaffer (1978) make an argument parallel to ours for the use of "multiplicative" evaluation functions.

Evans' algorithm required all figures to have the same number of parts, but was invariant under size, rotation, or translation changes. Ambiguities in evaluation were resolved by choosing rotation interpretations over reflections (in our formulation we choose "physical" interpretations over "economic" and "political" disputes).

An evaluation function of similar style was employed in the DENDRAL program (Buchanan and Feigenbaum, 1978). It would first eliminate candidate mass spectrometry readings which failed to meet certain constraints and then rank those that remain according to certain other rules of mass spectrometry. The DENDRAL evaluation function is described as follows:

> "MSPRUNE works with (a) a list of candidate structures from CONGEN, and (b) the mass spectrum of the unknown molecule. It uses a fairly simple theory of mass spectrometry to predict commonly expected fragmentations for each candidate structure. Predictions which deviate greatly from the observed spectrum are considered *prima facie* evidence of incorrectness; the corresponding structures are pruned from the list. MSRANK then uses more subtle rules of mass spectrometry to rank the remaining structures according to the number of predicted peaks found (and not found) in the observed data, weighted by measures of importance of the processes producing those peaks." (Buchanan and Feigenbaum, 1978)

One of the differences between the approach used by Evans and that used in DENDRAL is that additional domain knowledge is used to rank candidates in DENDRAL after the initial elimination phase. Evans uses a context-free pattern-matching process which considers all the available features of the problem descriptions. This type of context-free evaluation has been used by others since Evans (e.g., Winston, 1980; Carbonell, 1983). These approaches all advocate the selection of an analogy based on the accumulated evidence supporting matches over all possible features. It is not hard to imagine problems with rich representations whose object-to-object, feature-to-feature, and relation-to-relation comparisons for a reasonably large candidate set would make efficient identification of analogies extremely difficult. Winston (1980) demonstrates this problem by way of the illustration that there are N1!/(N1-N2)! ways to match two representations, where N1 is greater than or equal to N2. If N1 and N2 both have seven features there are 7! = 5040 alternative match possibilities just to characterize these two items alone. This is clearly not what we had in mind for retrieval performance. Constraining this type of matching requires some strong heuristics to direct or focus reasoning to the appropriate components of problems and permit similarity comparison primarily for critical features (Burstein, 1983; Buchanan and Feigenbaum, 1978).

6.6.2 Evaluation based on an invariance heirarchy

We have adopted an approach to evaluating the criticality of problem features based on a *relative-invariance hierarchy* among the feature components of problems (Carbonell, 1982). This invariance heirarchy of features has provided the guidance for both our choice of exclusion tests as well as the ranking of candidate cases. This invariance, which was derived by Carbonell from empirical analysis of metaphors, has been advocated as a cognitive model of people's expectations under various analogical transformations and is a more sophisticated method of deriving similarity than brute force pattern matching (Carbonell, 1983).

In essence, this approach says that object or disputant types should rarely be expected to match in analogies. They are the least preserved similarity. However, goal and planning knowledge is almost always preserved in analogies. This explains, for example, why Candy Dispute-0 is "similar" to Orange-Dispute-0 even though the actors and objects are all different. Both disputes involve actors whose goals are interpreted to be equivalent and have competitive goal relationships (Wilensky, 1983). Using the evidence that goal-related information is the most important feature for recall (Lichtenstein and Brewer, 1980) as well as analogy (Carbonell, 1982), we selected the goal relationship as one of the critical features for use as an elimination test for judging dispute similarity. Using the heuristic presented in Figure 3-16 to recognize goal relationships, Candy-Dispute-0 and Orange-Dispute-0 can be quickly classified as being equivalent in terms of goal relationship. The relative-invariance hierarchy provides a heuristic priority with which we order those aspects of a problem (dispute) which should be considered in evaluating similarity. Our interpretation of Carbonell's invariance hierarchy for the dispute domain is shown below:

THE MEDIATOR'S RELATIVE-INVARIANCE HIERARCHY FOR ANALOGIES

(1) Goals, goal relationship, and goal relationship derivation
(2) Argument type used by the disputants
(3) Physical, economic, or political dispute type
(4) Disputed object type
(5) Disputant type

Figure 6-16

The invariance hierarchy intuitively corresponds to a range of similarity evaluation, from the abstract goal and argument level to the more concrete object and disputant level. Goals and goal related information such as the goal relationship of a dispute are the highest rated features.

6.6.3 Eliminating cases based on goal derivations

In the domain of dispute mediation the goal relationship is frequently competitive. Thus, the discovery that a dispute has a competitive goal relationship will normally eliminate very few recalled cases. We, therefore, need some way to further differentiate among competitive dispute remindings. The solution to this problem is to identify and categorize the source of the competition. This is referred to as the *goal relationship derivation*.

Our approach is based on a classification of negative goal relationships developed by Wilensky (1983). Wilensky has identified three categories of negative goal relationships (negative goal relationships include both those goals that conflict between different agents as well as those that conflict within a single agent -- e.g., I want another piece of cake, but I also want to lose weight). Wilensky's categories are briefly defined below for the case when two different agents are in competition.

WILENSKY'S CLASSIFICATION OF NEGATIVE GOAL RELATIONSHIPS

1. Resource shortages - when two planners need the same consumable resource.

2. Mutually exclusive states - when two planners have goals that require exclusive states to exist at the same time.

3. Causing a preservation goal - when one planner's goals cause or threaten to cause the failure of another planner's goal.

Figure 6-17

We have seen numerous cases of competitive disputes, whose goal relationship was derived from a resource shortage. For example, Candy-Dispute-0 and the sea dispute mentioned earlier are examples of this type of competition. In both cases the disputants both wanted to use the same consumable object. In the MEDIATOR program, a non-renewable resource like minerals from the sea beds of the earth are considered equivalent to consumable objects since they are "consumed" when they are mined. Since these resource shortages seem to always occur in physical disputes, we have labeled this category "physical consumption derivations" in the program.

We have also seen cases that illustrate competition derived from mutually exclusive goal states of disputants. In Book-Dispute-0, for example, the students both wanted to check out the same book at the same time. The book is clearly not consumed, but both students cannot use it at the same time. When these mutually exclusive conflicts occur in physical disputes, we label them "physical exclusive competition" in the program.

Competition derived from the generation of a preservation goal can be illustrated by Solomon and the baby dispute. There were actually three different preservation goals active in that case. For the baby's real mother, a preservation goal (retaining control of her child) was caused when the second woman took her baby. Solomon also generated a preservation goal (preserve the life of her child) in the real mother when he threatened to divide the baby in half. For the second woman, when she lost her own child her desire to preserve her prestige caused her to steal the baby. Her action to satisfy this preservation goal thus caused the original competition. We have not used this category of negative goal relationship in investigating dispute derivations.

The MEDIATOR program uses the following heuristic algorithm to differentiate competitive cases based on goal relationship derivation.

MEDIATOR'S HEURISTIC FOR INFERRING GOAL RELATIONSHIP DERIVATION

1. When the dispute has a competitive goal relationship and either is a physical dispute or the dispute type is unknown

   and the disputed object is splittable and non-sharable and either consumable or non-renewable

   then infer that the dispute has a physical consumption derivation.

2. When the dispute has a competitive goal relationship and either is a physical dispute or the dispute type is unknown

   and the disputed object is either non-consumable or renewable

   then infer that the dispute has a physical exclusive competition.

Figure 6-18

This heuristic is demonstrated in the following excerpt from the MEDIATOR program. The program is considering the Antarctic dispute described at the beginning of this section. The MEDIATOR is reminded of three potentially applicable cases. All three cases have competitive goal relationships so this exclusion test fails to eliminate any of the cases. The goal relationship derivation as explained above eliminates all but one case in this situation.

MEDIATOR'S I/O BEHAVIOR USING GOAL RELATIONSHIP DERIVATION HEURISTIC

```
(negotiator antarctic-dispute t)
Considering the following dispute problem:
   fourteen nations are quarreling over the Antarctic,
   which has been presented as ako M-DISPUTE.
ATTEMPTING TO RECALL SIMILAR PROBLEMS IN ORDER TO CLASSIFY THIS ONE...
  looking for disputes with similar disputants or with similar goals...
  looking for disputes whose disputants used similar instrumental plans...
  looking for disputes involving similar objects...
   reminded of the case where Korea was in dispute
      because both disputants were of type M-POL-GROUP.
   reminded of the case where the minerals under the seas were in dispute
      because a disputant also had a goal of type M-EXTRACT ...
   reminded of the case where Korea was in dispute
      because the object in that case, KOREA,  was the same type object
      (M-LAND) as the Antarctic ...
   reminded of the case where two children are quarreling over candy
      because the argument used in that case, M-POSSESS,
      was the same type as this dispute.
There were three previous cases found.
   #<M-PHYS-DISPUTE 22332574> was the case where Korea was in dispute
   #<M-PHYS-DISPUTE 25067616> was the case where
      the minerals under the seas were in dispute
   #<M-PHYS-DISPUTE 22374716> was the case where
      two children are quarreling over a candy bar
There were three cases with the same COMPETITION goal relationship.
There was one case with the same PHYS-EXCLUSIVE-COMPETITION derivation.
   therefore Korea was in dispute is considered the best analogy to
      fourteen nations are quarreling over the Antarctic.
```

Figure 6-19

6.6.4 The MEDIATOR'S evaluation function

After all recalled cases have been compared to the current case according to the goal relationship and goal relationship derivation, there may still be many cases under consideration. We differentiate between any remaining candidate cases by assigning values weighted according to whether the recalled case matches the current case for all remaining features of the invariance hierarchy (i.e., argument type, disputed object type, or disputant type). This allows us to then sort all remaining candidate cases in terms of this "figure of

merit." If there is a clear winner after this ranking, that case is selected as the most analogous.

If two or more cases are equally rated after this ranking procedure, we then sort the highest rated cases according to the dispute type. We bias this sort to prefer physical disputes over economic disputes, which are in turn preferred over political dispute types. We then select the first case in the list returned from this sort as our preferred case.

The entire algorithm used by the MEDIATOR program to select the most appropriate case from a set of recalled cases for case-based reasoning is shown in Figure 6-20.

MEDIATOR'S EVALUATION FUNCTION FOR SELECTING APPROPRIATE CASES

1. Eliminate all cases with non-matching goal relationships.

   For each recalled case in the set of recalled cases
   when the recalled case has the same goal relationship
   as the current case
   collect the recalled case into a set of good cases.

2. Eliminate all cases with non-matching goal derivations.

   For each good case in the set of good cases
   when the good case has the same goal derivation as
   the current case
   collect the good case into a set of better cases.

3. Rate and order the better cases according to the three invariance features: arguments, disputed objects, and disputants;

   where similar arguments are awarded a value of 3,
   similar disputed objects have a value of 2, and
   similar disputants are awarded a value of 1.

4. If two or more cases are equally top rated, sort the equally rated cases such that the physical dispute cases are placed ahead of economic disputes, which are placed ahead of political disputes.

5. Return the first case from this sorted evaluated list as the most appropriate (i.e., most analogous) case.

Figure 6-20

6.7 Some implications

6.7.1 Problem solving and set effects

Shortly after integrating the conceptual long-term memory described above with the case based problem solving model, we began processing sequences of cases to test the behavior of the unified problem solver. We noticed that with a very few successful cases the MEDIATOR would quickly focus its reasoning and exclude other lines of reasoning. At first, we assumed this was a bug and began looking at our algorithms to try to "explain the failure." On further thought, we realized that this was a consequence of integrating a dynamic long-term memory with a case-based problem solving model. It meant that the MEDIATOR was exhibiting the behavior that Luchins (1942) had termed the "Einstellung effect" or "set effect."

Luchins had given subjects a series of "water-jug" experiments that required the subjects to use a set of jugs with various capacities and an unlimited supply of water to measure out a specified capacity. Luchins presented each subject with a series of problems, most of which could be solved using a simple sequence of addition and subtraction steps. What he observed during the presentation of his sequence of problems was that his subjects would quickly gravitate to a pattern as long as it was successful, even when there were alternate sequences that offerred shorter solutions.

6.7.2 Memory update and its effect on performance

Many AI problem solvers use a rule-based paradigm to perform their tasks (e.g., Hayes Roth et al., 1983). When these rules need to be changed, a system designer or domain specialist has to figure out where and how the rules need changing (Davis and Lenat, 1980). But once changed, the problem solver remains static in its knowledge application. A case based problem solver, in contrast, is designed to learn from and use its problem solving experience to dynamically alter its knowledge. This means that some of the knowledge and

processing that would normally be concentrated in the periodic modification of other static problem solvers needs to be included as part of the memory update process. Thus the total problem solving and memory update time for a case-based system will likely be longer than just the problem solving portion of strictly rule-based systems. The advantage, however, is in the partial automation of system optimization, incremental change in problem solving knowledge, and failure recovery available via the case-based approach.

The major portion of the knowledge necessary for memory maintenance is provided by the semantic domain model which used the instance and generalization primitives to describe the individual problem components. Just as this knowledge was important in index selection during retrieval, it is equally necessary in determining where a new case should be added to memory. Traversal in both instances is the same. However, the greatest value of these primitives is in the direction it provides for relating concepts to each other via generalization, as described below.

Using the approach of constructing a conceptual domain model, as described above, provides both a powerful aid to and a real limitation on what the system will learn. Its power comes from information about the domain that is external to individual cases. If the model accurately reflects the domain, then index selection and generalization is greatly simplified (e.g., recovery from a bad generalizations are minimized). Its limitation comes from the bias the model imposes on what the problem solver considers important. Thus concepts that are not included in the model are neither represented nor learned (Mitchell, 1981). No attempt has been made to dynamically detect and repair errors in the conceptual model. A possible future direction might be the application of case-based problem solving to the diagnosis and remediation of a problem solver's own conceptual model. The error recovery discussion presented in chapter five may be a step in that direction.

6.8 Summary

This chapter has presented our approach to organizing a conceptual memory of cases. We have sketched an integrated set of update and retrieval processes that define a process model of long term memory. In summarizing this model, a number of aspects bear review. First, conceptual memory management (update and retrieval processes), like other problem solving tasks, requires knowledge. We provide explicit knowledge in terms of a semantic heirarchy of instance and generalization concepts that correspond to the conceptual primitives used to model each domain component. This explicit knowledge is used to guide memory processes of generalization and index selection. This simplifies index selection and minimizes bad generalization, two problems that made previous conceptual memory models complex. Second, this model allows retrieval on partial matches, a capability not provided in other conceptual memory models (Kolodner, 1984). Third, retrieval has been simplified from previous models by the removal of elaboration. Elaboration is included as a component of the problem understanding process (described in chapter three). This distinction may be somewhat arbitrary, since our problem solving model allows multiple retrievals (one prior to understanding and one after). Elaboration is still an important part of the overall retrieval process because of the option of a post-elaboration retrieval and the fact that memory traversal does require index fitting, a type of elaboration.

The price for a simplifed model of conceptual memory is probably a loss of retrieval power in terms of available strategies for retrieval. The model also suffers a lack of flexibility to automatically add new generalization primitives. As implemented, conceptual memory records no frequency information, so frequency-based judgements are not now possible. There is no reason in principle why this information could not be included. In fact, the implemented M-MEMORY frames have an unused slot called norm-count which was originally intended to allow frequency-based judgements. The idea was that the norm-count would allow the problem solver to guage the certainty associated with a norm according to how many events it represented.

We have presented an evaluation function that is sensitive to the goal related features (goal relationship and goal relation derivation) that we believe are critical to making good selections from among several similar cases. This evaluation function, because it uses a series of elimination and ranking tests, avoids some of the problems with additive evaluation functions which tend to be insensitive to critical concept features.

CHAPTER VII

AN ANNOTATED EXAMPLE

In chapter one, we introduced the Sinai Dispute case for the first time and showed a simplified version of the MEDIATOR's I/O behavior on that case. At several points in subsequent chapters, we have used pieces of that case to illustrate other components of our problem solving model (see section 5.3.3 or 6.5.1, for example). The following long trace and annotation are provided to illustrate the MEDIATOR dealing with this dispute problem from start to finish. This also represents a summary of the extent to which case-based reasoning has been integrated into the problem solving process.

The "demand-driven" aspect of case-based reasoning (Schank and Birnbaum, 1980; Granger et al., 1984), which we referred to in chapter three, means that cases are retrieved from memory in response to a "demand" by the problem solving process for help in making a choice or decision. By the same token, when the problem solving process has sufficient knowledge with which to reason, there is no requirement for a memory retrieval. This is evident in two different contexts in this longer trace. When the MEDIATOR already has a case from which to reason, such as during the first planning pass, the program does not need to retrieve cases from memory so, without demand, a memory retrieval is avoided. Second, after the failure has been remedied, the program already has a complete representation of the dispute so the demand for a memory retrieval is absent once again.

We first repeat a text version of the case to refresh memories:

SINAI DISPUTE

A mother reads in the paper about the Sinai dispute (before the Camp David Accords). She is reminded of the Korean War since both are disputes over land, both are competitive situations in which the conflict cannot be resolved completely for both sides, and in both, military force had been used previous to negotiations. Based on this reminding, she predicts that Israel and Egypt will divide the Sinai equally, since that is what happened in the Korean War.

She later reads that the USA had suggested this solution and it had been rejected by both sides. She is reminded of her daughters' quarrel over an orange. She had suggested that they divide it equally, and they had rejected that, since one wanted to use the entire peel for a cake. Realizing that she hadn't taken their real goals into account, she then suggested that they "divide it into different parts" -- one taking the peel, the other the fruit. This reminding provides the suggestion that failures may occur because the goals of the disputants are misunderstood. She therefore attempts a reinterpretation of Israel's and Egypt's goals. By reading more closely, she learns that Israel wants the Sinai as a military buffer zone in support of national security, and Egypt wants the land back for national integrity.

She is reminded of the Panama Canal dispute since the disputants, disputed object, and goals are similar to those in the newly understood Sinai dispute. In that case, the USA returned economic and political control of the Canal to Panama, but retained military control for national security reasons. Analogy to that incident leads the mother to decide that a similar division of the Sinai would be reasonable and guides the refinement of the "divide into different parts" plan. Replacing the US by Israel (the party currently in control of the object) and Panama by Egypt (the party who used to own it and wants it back), she predicts that Egypt will get economic and political control of the Sinai, while its normal right of military control will be denied.

The MEDIATOR is told to suggest a resolution to the Sinai Dispute. The initial representation of the dispute as presented to the program is a frame of type M-DISPUTE. A list form of the frame is produced and displayed below.

I/O BEHAVIOR SHOWING CASE-BASED PROBLEM SOLVING IN THE SINAI DISPUTE

```
(mediator sinai-dispute t)
Considering the following problem:
   Israel and Egypt both want the Sinai,
   which has been presented as ako M-DISPUTE.
```

```
(*DISPUTE*
 (PARTY-A (ISRAEL))
 (PARTY-B (EGYPT))
 (DISPUTED-OBJ SINAI)
 (ARGUMENT-A (*ARGUMENT* (ARGUER (ISRAEL))
                         (SUPPORT (*PHYS-CONTROL* (ACTOR ISRAEL)
                                                  (OBJECT SINAI)
                                                  (INST *MILITARY*)))))
 (ARGUMENT-B (*ARGUMENT* (ARGUER (EGYPT))
                         (SUPPORT (*PHYS-CONTROL* (ACTOR EGYPT)
                                                  (OBJECT SINAI)
                                                  (INST *MILITARY*))))))
```

****

As the MEDIATOR begins to interpret the presented problem, it attempts to specialize the dispute by classifiying it into one of its known dispute types. To support this decision, the case-based reasoning process attempts to provide appropriate exemplars. Components of the initial representation of the dispute are used to identify the generalized episodes to be traversed. Using the appropriate components (e.g., the disputed object) as target cues, traversal procedures locate and retrieve the most similar previous case from generalized episodes corresponding to each of the components of the dispute (see section 6.5.1). For the Sinai Dispute, the generalized episodes associated with the disputants, disputed object, and disputant argument are probed. Because the disputants' goals are not explicitly presented, no reminding can be attempted based on goals.

****

```
ATTEMPTING TO RECALL SIMILAR PROBLEMS IN ORDER TO CLASSIFY THIS ONE...
  looking for disputes with similar disputants or with similar goals...
  looking for disputes whose disputants used similar arguments.
  looking for disputes involving similar objects...
   reminded of the US and Panama are quarreling over the Canal
      because both disputants were of type M-POLITY.
   reminded of Korea was in dispute
      because the object in that case, KOREA,
      was the same type object (M-LAND) as SINAI and
      because the argument used in that case, the military force argument,
                           was of the same type, M-MILITARY-FORCE, as this dispute.
```

****

Two cases were retrieved (out of the four in its memory at the time) as potentially applicable sources for knowledge transfer. The program then uses its evaluation criteria to judge the appropriateness of each case (see section 6.6.4). It selects the Korean Conflict as the most appropriate case because it shared both object and argument similarity and thus received a higher rating. Focussing on this case, it then transfers the classification from that case since no constraints are violated. This done, the dispute can now be reinstantiated as a physical dispute (see section 3.3).

****

```
Using default evaluation function, ranking cases based on
   the three invariance features dispute arguments, objects,
   and disputants.
Selected the dispute where Korea was in dispute
   which was classified as M-PHYS-DISPUTE for further consideration.
Transferring previous classification to this dispute.

The current dispute will be referred to as #<M-PHYS-DISPUTE 40533552>
```

****

Having classified the dispute, the program next begins to elaborate the dispute representation (see section 3.4). At this point, the MEDIATOR notices that the dispute representation lacks goal information for the disputants. The importance of goal information is implicit in the MEDIATOR's algorithms. It's heuristics direct it to first consider inferring the goals from information given directly in the initial representation. Thus, it tries to infer the goals

from the disputants' arguments, based on a belief that disputants often indicate their intentions in their persuasive arguments (see section 3.4.i). The program is currently biased to prohibit any such inference based on coercive arguments, such as the use of physical force (see section 2.3.4). It then goes to its next best source of inference, the previously recalled case (in this instance the Korean Conflict). Since the goals in that case are consistent with what is known about the Sinai dispute (see section 3.4.3), they are transferred and instantiated for the current case. The goals of Egypt and Israel are thus inferred to be physical control over the Sinai. Once the goals have been inferred, the program can decide the goal relationship (see section 3.4.4) here the relationship is competition.

****

ISRAEL and EGYPT have both presented arguments recognized as
  type *PHYS-CONTROL* which is normally presented in an
  attempt to persuade by force.
Therefore no inferences will be based on these arguments.

Attempting to transfer goal type from recalled case
   #<M-PHYS-DISPUTE 40306114>
   checking for consistency with normal uses of SINAI.
Thus ISRAEL and EGYPT are both inferred to have a M-PHYSICAL-CONTROL goal
   which is consistent with the normal uses of SINAI in this context.

Goal relationship is COMPETITION.

****

The MEDIATOR has now completed the process of problem understanding. Next the program begins the planning process (see chapter four). Because a case was retrieved during the understanding process, there is no need to probe memory again since that same case is still available for case-based reasoning.

****

ATTEMPTING TO SELECT A MEDIATION PLAN
   TO RESOLVE THE DISPUTE IDENTIFIED AS #<M-PHYS-DISPUTE 40533552>.
Using previously recalled case, where Korea was in dispute.
   It was resolved using the plan known as divide equally.
Checking for applicability of that plan to the current case..
My reasoning is as follows:
   This plan has not failed to my knowledge on similar previous
   problems, and it normally doesn't make sense to share SINAI,
   SINAI can be divided without loss of functionality;
   when this is considered with a compromise planning policy
   and my inference that the parties' goals are in competition;
   all indicate that divide equally is a reasonable plan.
Selecting the plan divide equally for this dispute and instantiating.
I suggest that the plan called divide equally be used.

****

Because the problem solver needed to select a plan to resolve this dispute, the recalled case is examined to determine if transfer of the planning policy and plan to the current case might be appropriate. The preconditions associated with the recalled plan ("divide equally" in this instance) are examined for the current case. The preconditions provide both an indication of the plan's acceptability and an explanation of its decision (see section 4.4.3). Since this plan's precondition tests are satisfactory for this case, it is accepted, and "divide equally" becomes the selected abstract plan (see section 4.4). Next, the MEDIATOR attempts to specialize the abstract "divide equally" plan into one of its known specializations. It does this by default reasoning since the recalled case can offer no more help (see section 4.5.2).

****

```
TRYING TO SPECIALIZE DIVIDE EQUALLY BY DEFAULT REASONING.
Looking at the plan called "taking turns"
   which does not seem applicable.
Looking at the plan called "split the difference"
   which does not seem applicable.
Considering the plan called "one cuts the other chooses"
   which appears to be applicable.

My additional reasoning is as follows:
   The fact that SINAI can be split without destruction
   and my classification of this dispute as a M-PHYS-DISPUTE
   indicate to me that "one cuts the other chooses" is a reasonable plan.

I suggest that the plan called "one cuts the other chooses" be used.
INSTANTIATING EXPECTED CONTRACT RESULTING FROM PLAN APPLICATION.
     ************************************************************************
```

The appropriate specialization of the abstract plan is selected by testing the preconditions of each alternative in turn. The first specialized plan whose preconditions are true is selected. The explanation provided after plan refinement is produced in the same manner as described in section 4.4.3.

The MEDIATOR next examines its previous experience with the selected plan in order to reconstruct a composite contract that represents all previous uses of this plan (see section 4.5.2). The composite contract is judged acceptable according to a heuristic evaluation function much like that used to evaluate cases retrieved from memory (see section 6.5.2). If the composite is acceptable, then expectations for specific events and portions of the composite can be used to guide the instantiation of the new contract. Basically, the predictions from the plans describe how the goals of the disputants are likely to be realized as the result of the mediation process. In this case, the contract predicts that each polity will take physical control of half of the Sinai.

```
     ************************************************************************

ATTEMPTING TO RECALL PREVIOUS EXPERIENCE WITH ONE CUTS THE OTHER CHOOSES
  reconstructing my previous experience with this plan
     results in a contract identified as #<M-DIVIDED-OBJ-CONTRACT 12475742>
  checking the applicability of this contract to this situation ...

Evaluating old contract based on the four invariance
   features disputant goals, dispute type, disputed object, and disputants.

With a rating of 15 out of 17,
   the old contract is judged acceptable based on these criteria.

  using it to guide current contract construction ...
   matching ISRAEL with CHILD1 ...
   matching EGYPT with CHILD2...
   matching SINAI with CANDY1...
   matching
(*GOAL* (*PHYS-CONTROL* (ACTOR ISRAEL) (OBJECT SINAI)))T with
(*HALF* CANDY1)T...
   matching
(*GOAL* (*PHYS-CONTROL* (ACTOR EGYPT) (OBJECT SINAI)))T with
(*HALF* CANDY1)T...
   transferring other components of contract unchanged.
PREDICTIONS ASSOCIATED WITH USING ONE CUTS THE OTHER CHOOSES FOR THE
  CURRENT DISPUTE KNOWN AS #<M-PHYS-DISPUTE 40533552> ARE EMBODIED IN THE
  CONTRACT KNOWN AS #<M-DIVIDED-OBJ-CONTRACT 16232557>
     ************************************************************************
```

Having made its recommendation, the program next seeks feedback in order to evaluate its efforts. Our feedback takes two forms. We initally signal our rejection via a negative response to an explicit request for evaluation. In addition, we provide a conceptual representation that is intended to provide some clues to aid in directing error recovery. The representation below stands for the situation where the disputants explicitly express previously unstated goals. The command "external-intentional-direct" is the atom for the repreentation below it. That representation tells the program that the failure was the result of "external" evaluation, that

the failure was a goal failure (i.e., a failure of intentions), and that the feedback provides the appropriate direction. In this situation, the problem solver has to interpret the feedback during blame assignment and apply the appropriate remedy. To accomplish this, the MEDIATOR begins a new problem solving cycle.

****

```
Do you agree, that this is the best solution? (Y or N) No.
   **** ONE CUTS THE OTHER CHOOSES not acceptable ****

Can you provide any comments that might help me remedy this failure?
external-intentional-direct
You said:
((*MTRANS* (ACTOR ISRAEL)
           (MOBJECT (*GOAL* (*NAT-SECURITY* (ACTOR ISRAEL)
                                            (OBJECT SINAI)))))
 (*MTRANS* (ACTOR EGYPT)
           (MOBJECT (*GOAL* (*NAT-INTEGRITY* (ACTOR EGYPT)
                                             (OBJECT SINAI))))))
Attempting to explain this failure and find a new solution.
Considering the following problem:
   failed mediation for Israel and Egypt both want the Sinai,
   which has been presented as ako M-UNSUCCESSFUL-MEDIATION.
   It will be referred to as #<M-UNSUCCESSFUL-MEDIATION 40544074>
ATTEMPTING TO RECALL SIMILAR PROBLEMS IN ORDER TO CLASSIFY THIS ONE...
ATTEMPTING TO RECALL SIMILAR FAILURES ...
  looking for previous mediation plan failures...
  looking for failures with similar disputants or with similar goals..
  looking for failures involving similar objects...
   reminded of the failed mediation for two sisters are quarreling over
      an orange for which the plan "one cuts the other chooses" also
      failed and because the object in that case, ORANGE1, and SINAI
      are both of type M-PHYS-OBJ.
There was one previous case found.
   #<M-WRONG-GOAL-INFERENCE 5304703> was the
   failed mediation for two sisters are quarreling over an orange.
Failure in that case was because of M-WRONG-GOAL-INFERENCE.
Transferring that classification to this failure.
The current failure will be referred to as #<M-WRONG-GOAL-INFERENCE 40544535>
```

****

During this new problem solving cycle, the MEDIATOR has used the features of this failed mediation attempt to probe for previous similar failures. As evidenced by the output, there was one previous failure that shared the same mediation plan failure, namely Orange-Dispute-0. Blame for that failure was ascribed to a wrong goal inference during problem understanding. Since the current case also involved goal inferences, this type of failure is transferred as a working classification of the failure in this situation (see section 5.4.4). Next the program tries to find a remedy for the knowledge that led to this failure.

****

```
ATTEMPTING TO SELECT A REMEDY FOR
   THE FAILURE IDENTIFIED AS #<M-WRONG-GOAL-INFERENCE 40544535>.
Using previously recalled case,
                        failed mediation for two sisters are quarreling over an orange.
   It was corrected using the remedy known as M-USE-ACTUAL-EVENTS.
Checking for applicability of that plan to the current case..
Unable to use that remedy for this failure because
                        the known preconditions are not satisfied.
Using default reasoning to select a resolution plan.
Considering remedies normally useful for M-WRONG-GOAL-INFERENCE failures.
Looking at the remedy called infer goal from response
                        which appears to be applicable.
Using feedback to consider other possible goal interpretations
   to explain my failure in this situation ...
I previously inferred that both ISRAEL and EGYPT had M-PHYSICAL-CONTROL
   type goals.
Based on the feedback, I will replace ISRAEL's goal with
                        a M-NATIONAL-SECURITY type goal and EGYPT's goal with
                        a M-NATIONAL-INTEGRITY type goal.
Both ISRAEL's and EGYPT's goal representation are now changed.
```

```
Remediation complete.
     ************************************************************************
```

Using this interpretation of the failure, the MEDIATOR attempts to transfer the remedy used to recover from the previous failure. In the previous case, the MEDIATOR had used the remedy known as USE-ACTUAL-EVENTS. This remedy is appropriate when the feedback is in terms of a report of an attempted plan execution. Since the feedback in this case does not contain actual execution events, the plan is inappropriate. Because the feedback was an explicit indication of the disputants' goals, the appropriate remedy is to transfer the goals from the response. With this change, the MEDIATOR exits the remediation cycle and reconsiders the new representation of the original problem.

```
     ************************************************************************

Given this new information, I'll reconsider this problem.
Considering the following problem:
   Israel and Egypt both want the Sinai,
   which has been presented as ako M-PHYS-DISPUTE.
(*PHYS-DISPUTE*
 (PARTY-A (ISRAEL (*GOAL* (*NAT-SECURITY* (ACTOR ISRAEL)
                                          (OBJECT SINAI)))))
 (PARTY-B (ISRAEL (*GOAL* (*NAT-INTEGRITY* (ACTOR EGYPT)
                                           (OBJECT SINAI)))))
 (DISPUTED-OBJ SINAI)
 (ARGUMENT-A
  (*ARGUMENT*
   (ARGUER (ISRAEL (*GOAL* (*NAT-SECURITY* (ACTOR ISRAEL)
                                           (OBJECT SINAI)))))
   (SUPPORT (*PHYS-CONTROL* (ACTOR ISRAEL) (OBJECT SINAI)
 (ARGUMENT-B
  (*ARGUMENT*
   (ARGUER (EGYPT (*GOAL* (*NAT-INTEGRITY* (ACTOR EGYPT)
                                           (OBJECT SINAI)))))
   (SUPPORT (*PHYS-CONTROL* (ACTOR EGYPT) (OBJECT SINAI)
                                          (INST *MILITARY*)))))
This dispute will be referred to as #<M-PHYS-DISPUTE 40533552>
Goal relationship is CONCORDANT.
     ************************************************************************
```

Notice that during this second pass through the problem understanding phase of problem solving, the MEDIATOR does not attempt to retrieve any previous experience. This is not because the earlier case has been retained (as a matter of fact the old case is discarded as a result of the failure), but results from the fact that the problem representation has already been elaborated during the previous attempt so there is no need. As a consequence of the change in the disputants' goal representation during remediation, the dispute is now classified as having a concordant goal relationship. This new knowledge will significantly influence the evaluation of previous cases retrieved in support of plan selection, as shown below.

```
     ************************************************************************

ATTEMPTING TO SELECT A MEDIATION PLAN
   TO RESOLVE THE DISPUTE IDENTIFIED AS #<M-PHYS-DISPUTE 40533552>.
  looking for disputes with similar disputants or with similar goals...
  looking for disputes whose disputants used similar arguments.
  looking for disputes involving similar objects...
   reminded of the US and Panama are quarreling over the Canal
      because both disputants were of type M-POLITY.
   reminded of Korea was in dispute
      because the object in that case, KOREA,
      was the same type object (M-LAND) as SINAI and
      because the argument used in that case, the military force argument,
                              was of the same type, M-MILITARY-FORCE, as this dispute.
```

```
There was one previous case with the same CONCORDANT goal relationship.
   The US and Panama are quarreling over the Canal is considered
   the most analogous case to Israel and Egypt both want the Sinai.
   It was resolved using the plan known as "divide into different parts".
Checking for applicability of that plan to the current case..
My reasoning is as follows:
   This plan has not failed to my knowledge on similar previous problems
   SINAI can be divided without destruction and when this is considered
   with my initial classification of this dispute as a M-PHYS-DISPUTE
   and my inference that the parties' goals are concordant all indicate
   to me that division into different parts will satisfy everyone.
Selecting the plan "divide into different parts"
                          for this dispute and instantiating.
I suggest that the plan called divide into different parts be used.
     **************************************************************************
```

Because the disputant's representations have been elaborated with goals and a goal relationship has been inferred for the dispute since the previous memory retrieval, the process of evaluating the most applicable case (see section 6.5.4) is able to determine that the Panama Canal dispute is more applicable. This is because we treat goal related similarities as more important criteria. As a result of the new case selection, a different plan, "divide into different parts," is identified and determined to be applicable. Notice that because this plan is already a specialized plan, there is no need to perform the specialization step of plan refinement. The program next sets about to instantiate the selected plan. The primary effort is to instantiate the contract. This is done by reconstructing a contract based on previous use of that type of contract (see section 4.5.2).

```
     **************************************************************************

INSTANTIATING EXPECTED CONTRACT RESULTING FROM PLAN APPLICATION.
ATTEMPTING TO RECALL PREVIOUS EXPERIENCE WITH DIVIDE INTO DIFFERENT PARTS
  reconstructing my previous experience with this plan
     results in a contract identified as #<M-DIFF-PARTS-CONTRACT 40343224>
  checking the applicability of this contract to this situation ...
Evaluating old contract based on the four invariance
   features disputant goals, dispute type, disputed object, and disputants.

With a rating of 17 out of 17,
   the old contract is judged acceptable based on these criteria.
  using it to guide current contract construction ...
   matching ISRAEL with USA ...
   matching EGYPT with Panama...
   matching SINAI with Panama Canal...
   matching
(*GOAL* (*NAT-SECURITY* (ACTOR ISRAEL)
                        (OBJECT SINAI))) with
(*GOAL* (*MIL-CONTROL* (ACTOR USA)
                       (OBJECT "Panama Canal")))...
   matching
(*GOAL* (*NAT-INTEGRITY* (ACTOR EGYPT)
                         (OBJECT SINAI))) with
(*GOAL* (*POLITICAL-CONTROL* (ACTOR  PANAMA)
                             (OBJECT "Panama Canal")))...
   transferring other components of contract unchanged.
PREDICTIONS ASSOCIATED WITH USING DIVIDE INTO DIFFERENT PARTS FOR THE
  CURRENT DISPUTE KNOWN AS #<M-PHYS-DISPUTE 40533552> ARE EMBODIED IN THE
  CONTRACT KNOWN AS #<M-DIFF-PARTS-CONTRACT 40547646>
Do you agree, that this is the best solution? (Y or N) Yes.
Do you know the results of the plan's execution? (Y or N) Yes.

Please indicate the results: results-ok
     **************************************************************************
```

Based on the new interpretation of the dispute, the MEDIATOR has decided that the dispute between the US and Panama is now the more appropriate exemplar to the current case. As a consequence, the "divide into different parts" plan is evaluated and selected for the current case in order to effect an agreeable division. The expectations associated with the employment of this plan are that Egypt will get political control of the Sinai in order to salvage its

national integrity, without retaining its normal right of military contol which is in essence what resulted from the Camp David Accords.

****

```
There was one previous case with the same CONCORDANT goal relationship.
   The US and Panama are quarreling over the Canal is considered
   the most analogous case to Israel and Egypt both want the Sinai.
   It was resolved using the plan known as "divide into different parts".
Checking for applicability of that plan to the current case..
My reasoning is as follows:
   This plan has not failed to my knowledge on similar previous problems
   SINAI can be divided without destruction and when this is considered
   with my initial classification of this dispute as a M-PHYS-DISPUTE
   and my inference that the parties' goals are concordant all indicate
   to me that division into different parts will satisfy everyone.
Selecting the plan "divide into different parts"
                        for this dispute and instantiating.
I suggest that the plan called divide into different parts be used.
     **************************************************************************
```

Because the disputant's representations have been elaborated with goals and a goal relationship has been inferred for the dispute since the previous memory retrieval, the process of evaluating the most applicable case (see section 6.5.4) is able to determine that the Panama Canal dispute is more applicable. This is because we treat goal related similarities as more important criteria. As a result of the new case selection, a different plan, "divide into different parts," is identified and determined to be applicable. Notice that because this plan is already a specialized plan, there is no need to perform the specialization step of plan refinement. The program next sets about to instantiate the selected plan. The primary effort is to instantiate the contract. This is done by reconstructing a contract based on previous use of that type of contract (see section 4.5.2).

```
     **************************************************************************

INSTANTIATING EXPECTED CONTRACT RESULTING FROM PLAN APPLICATION.
ATTEMPTING TO RECALL PREVIOUS EXPERIENCE WITH OIVIDE INTO OIFFERENT PARTS
  reconstructing my previous experience with this plan
     results in a contract identified as #<M-OIFF-PARTS-CONTRACT 40343224>
  checking the applicability of this contract to this situation ...
Evaluating old contract based on the four invariance
   features disputant goals, dispute type, disputed object, and disputants.

With a rating of 17 out of 17,
   the old contract is judged acceptable based on these criteria.
  using it to guide current contract construction ...
   matching ISRAEL with USA ...
   matching EGYPT with Panama...
   matching SINAI with Panama Canal...
   matching
(*GOAL* (*NAT-SECURITY* (ACTOR ISRAEL)
                        (OBJECT SINAI))) with
(*GOAL* (*MIL-CONTROL* (ACTOR USA)
                       (OBJECT "Panama Canal")))...
   matching
(*GOAL* (*NAT-INTEGRITY* (ACTOR EGYPT)
                         (OBJECT SINAI))) with
(*GOAL* (*POLITICAL-CONTROL* (ACTOR  PANAMA)
                             (OBJECT "Panama Canal")))...
   transferring other components of contract unchanged.
PREOICTIONS ASSOCIATEO WITH USING OIVIOE INTO OIFFERENT PARTS FOR THE
  CURRENT OISPUTE KNOWN AS #<M-PHYS-OISPUTE 40533552> ARE EMBOOIEO IN THE
  CONTRACT KNOWN AS #<M-OIFF-PARTS-CONTRACT 40547646>
Oo you agree, that this is the best solution? (Y or N) Yes.
Oo you know the results of the plan's execution? (Y or N) Yes.

Please indicate the results: results-ok
     **************************************************************************
```

Based on the new interpretation of the dispute, the MEDIATOR has decided that the dispute between the US and Panama is now the more appropriate exemplar to the current case. As a consequence, the "divide into different parts" plan is evaluated and selected for the current case in order to effect an agreeable division. The expectations associated with the employment of this plan are that Egypt will get political control of the Sinai in order to salvage its

national integrity, without retaining its normal right of military contol which is in essence what resulted from the Camp David Accords.

****

7.1 MEDIATOR implementation details

The MEDIATOR is implemented as an object-oriented program in ZETALISP using Flavors (Weinreb and Moon, 1981) which runs on either a Symbolics 3600 or 3670 processor. Its semantic knowledge is described in terms of 154 different Flavor definitions. The MEDIATOR exists as three files: a Flavors file that contains its semantic knowledge and some procedural knowledge in terms of 69 different Flavor methods, a file of ZETALISP functions which provide its implicit problem solving model, and a data file that provides 261 initial "object" instances and other explicit knowledge provided a priori to the program (i.e., its beginning episodic knowledge). The total file size of all Flavor and method definitions is 120,000 characters. The functions file is approximately 145,000 characters and the data file is approximately 34,000 characters.

Since the MEDIATOR is an experimental prototype and our interest was in looking at problem solving in an integrated fashion, no attempt was made to push its implementation to extremes in any one direction. For example, the knowledge necessary to perform some part of a successful mediation, using at least one line of reasoning, was implemented for only 14 different cases. On the other hand, extensive knowledge was implemented to demonstrate many varieties of the candy and orange disputes. These two cases were used to "explore" variability within the space of one case scenario. The Sinai dispute, on the other hand, was used to push the program "vertically" into more of a end-to-end integrated system. With all the different combinations of initial representations, elaborated understandings, planning selections, and failure options the number of different implemented "cases" is on the order of 50 or even more if you choose a strict definition of "different."

At its largest, there were nine cases in the MEDIATOR's episodic memory. At that point, we were seriously cramping the memory management scheme of the Symbolics and garbage collection became the dominant activity. No further attempts were made to either restrict update or find ways around default swapping memory limitations.

In terms of conceptual memory performance, the length of time required to update memory grows rapidly with each additional case because of the paging involved when using only one megabyte of real memory. However, the required update time seemed to level off after about six cases. There was no attempt to maintain strict time measurements, but update at its worst required on the order of thirty minutes. As we discussed in chapter six, update time can be sacrificed for retrieval time. Even with nine cases in memory, retrieval was quite good typically on the order of a few seconds. These performance characteristics were achieved without any real effort toward efficiency (e.g., often the code was run uncompiled as a concession to ease of debugging). Other than memory update time, the program handles the Sinai dispute case, as shown above, in less than five minutes (depending on how fast you respond to its requests for feedback).

CHAPTER VIII

CONCLUSIONS AND SOME COMPARISONS

> "In order to solve a new problem one uses what might be called the basic learning heuristic - first try using methods similar to those which have worked, in the past, on similiar problems." (Minsky, 1963)

8.1 Conclusions

Human problem solvers are confronted with new and difficult problems everyday. In dealing with new problems, people seem to be able to bring the appropriate knowledge from their past experience to bear on the current problem. Case-based reasoning is a computational model of this process. It implies that a problem solver can become more effective by increasing its episodic knowledge, organizing this knowledge so that it can be made available when needed, and knowing how to transfer the applicable portions from past experience.

We have presented a computer process model of problem solving that shows how the case based process can make a problem solver more effective. In each stage of problem solving, from problem understanding, plan generation, and results evaluation, case-based reasoning seems to offers a means to help computer systems adapt to changes in its environment. As we have shown, successes cause a problem solver to adopt similar lines of reasoning again, while failures cause those decisions to be avoided. In this way a problem solver can adapt quickly to the demands of the environment.

Case-based reasoning requires finding previous cases in memory, recognizing which might be applicable to the new situation, and transferring appropriate components of the previous case to the new one. It is this last step that we have called "analogical transfer." Our chapters have given details of how this transfer is done in each of the major phases of problem solving. In general, the transfer process can be described as "demand driven." This means that cases are retrieved in response to a "demand" on the problem solving process for help in making a choice or decision. When a problem solver believes there is sufficient knowledge then there is no "demand" made on memory.

In the rest of this section, we offer three other pieces of evidence in support of the case-based model of problem solving. First, we will present some psychological evidence that provides at least plausible support for the model. Second, we will describe why we believe case-based reasoning will be generalizable to other domains and under what circumstances it should prove an effective technique. Finally, we will walk through a very simple analysis that summarizes why we expect case-based reasoning will require less reasoning than default processing of problems. Because of the advantages case-based reasoning can offer in certain circumstances, we are convinced that it should be recognized as an important paradigm for problem solving.

8.1.1 Psychological validity

We have been guided in the development of the case-based model of problem solving by a sizable body of empirical work reporting on various psychological facets of human problem solving in diverse contexts and at various stages of expertise. In particular, we have attempted to maintain consistency between our computation design decisions and those unambiguous aspects of this literature on human problem solving. While it was not our objective, we do believe there are potential insights to be gained by the study of our model as a computational theory of human problem solving and learning. In this section, we will discuss evidence in several areas that provide the "inherited" consistency that we have attempted to maintain. Specifically, the areas we will mention include analogical problem solving and problem classification.

Analogical problem solving - One of the best and earliest known investigations of analogical problem solving in an unusual context (i.e., "water jug" problems) is the work of Luchins (1942). He showed how subjects would persist in the use of a previous problem strategy ("plan") for similar problems even when they could be solved more simply by other strategies. This has been labeled the "set effect" or "Einstellung effect." Although not investigated extensively, the case-based model of problem solving represents a computational explanation for this behavior. The MEDIATOR, for example, will cling to a specific mediation plan that has been successful in the past if it believes the new case is similar to an earlier case. In one sense, we are encouraged by behavior that is consistent with known psychological evidence, but this behavior can also be viewed as inhibiting a problem solver from easier solutions. With further research we may better understand the role of analogical reasoning in

problem solving, so that we must be able to distinguish between the useful and inhibiting roles of analogy. Even though people seem inclined to exhibit "set effects," we may not want computer problem solvers to be so inclined. There are sure to be situations where computer systems will need to use the knowledge of "set effects" to determine its behavior. For example, systems that are employed as "tutors" will find the knowledge of this tendancy important in following and modelling the reasoning of a human student and as a guide to the appropriate selection of test problem sequences.

Other psychological studies of analogical problem solving in unusual problem contexts (e.g., missionaries and cannibals by Reed et al., (1974), Tower of Hanoi by Hayes and Simon (1979), and the "radiation" problem by Gick and Holyoak (1980)), suggests that naive problem solvers have considerable difficulty in transferring problem solving strategies between "semantically distant" but structurally isomorphic versions of the same problems. On the surface, these studies might seem to contradict our assumption that analogical reasoning (and case-based reasoning) are crucial to effective problem solving. However, later studies by Gick and Holyoak (1983), aimed at facilitating analogical transfer between related problems, supports our model in that they claim that analogical transfer is organized around learned generalized problem classes or "schemas." The "schemas" described by Gick and Holyoak correspond nicely to our problem classes, e.g., "physical disputes." Their work also tends to confirm the focus provided by these classes in directing the transfer of components from recalled cases.

Studies of transfer between analogously related problems mentioned above have been conducted with relatively naive subjects. This perhaps accounts for the apparent lack of contribution by analogical reasoning to the problem solving process. According to our model this deficit could be explained as resulting from the fact that novice reasoners draw primarily from literally similar problem solving cases, since their case repertoire of potential analogs is limited (Chi, et al, 1981; Ross, 1982). With more experience, the case-based model of problem solving predicts that these subjects will use analogical reasoning more extensively. Several researchers (Chi, et al., 1981; Clements, 1981, 1982) have suggested this and further argue that experts tend to use chains of related analogies which are both drawn from experience and generated by changing components of the original problem.. The process models suggested separately by Ross and Clements have both suggested that analogy fits into a general problem solving context, both for understanding and solution planning. They do not, however, provide the information processing detail that explains how this is accomplished as we have provided here. For example, they do not specify how cases are organized in memory, nor provide algorithms that explain why a particular case is recalled and how it is used.

Classification and problem solving - We have argued that problem understanding, which includes classification into known problem types, is an essential stage of the problem solving process (Greeno, 1977). The classification stage of understanding prepares a problem solver to augment the representation with more specific domain knowledge. This process has been observed by several researchers (Chi, et al., 1981; Clement, 1981, 1982; Hayes and Simon, 1979; Hinsley et al., 1977; Paige and Simon, 1979; Mayer et al., 1984). The case-based model of problem solving explicitly recognizes the classification and elaboration phases as important stages of problem understanding. One example of the evidence supporting this approach is the work of Hinsley et al. (1977) who investigated algebra word problems. This study showed that subjects can reliably identify problem types, that classification occurs early in understanding the problem (after 18% of the text has been read), that subjects can accurately predict what kind of information will appear later in the problem, and that the subjects can state known plans which will prove effective even before reading all the problem statement. This type of study shows that our attachment of "plans" directly to the problem types as explained in chapter three and four is consistent with empirical evidence.

Other experiments by Hinsley et al. (1977) designed to "confound" this problem classification process showed that subjects attempt to apply "plans" even in the absence of confirmatory evidence (a kind of set effect that is consistent with our problem solving model). Subjects were presented with semantically nonsensical cover stories which led some of them to incorrectly classify problems based on irrelevent context cues. These subjects then would attempt "plans" consistent with their inappropriate interpretation despite the absence of additional evidence. In another study, Mayer et al. (1984) found that error recall rates were higher for problem components that were irrelevant to the underlying problem type. This supports our assertion that knowledge of problem classes is an important element in organizing the problem components into a coherent representation during understanding. These findings are consistent with the earlier results of Chi et al. (1981) who found the same behavior exhibited by expert physics problem solvers. They also tended to categorize problems in terms of problem classes related to underlying physical processes as opposed to surface features. These underlying physical processes were called "derived features" by Chi et al. When expert physics problem solvers recognized these "derived features" there seemed to be immediate access to specific class-dependent "plans." The use of known problem classes as a major contributor in the acquisition of problem solving skills is thus consistent with existing psychological evidence.

Our model of problem solving provides an information processing explanation for, and a classification of, the errors human problem solvers commit. For example, our model predicts that problem solvers who lack the knowledge necessary to relate separate parts of problems into a coherent problem "frame" will have difficulty in understanding problems correctly. This was found to be the case when Mayer et al. (1984) analyzed the errors in story problem recall by college students and found that the relational portions of problem statements were

most difficult to recall and resulted in poor performance. In addition, our model of failure recovery can be used to explain how experts use analogy to previous failures to diagnose and remedy failures in steam-plants or other systems (Rouse, 1983).

8.1.2 Generalizing to other domains

In demonstrating the advantages of case-based reasoning in problem solving, our examples have been drawn extensively from the task domain of common sense dispute mediation. This dependance on a single task domain could raise questions of generality across different domains. In this section, I will present a discussion of some general characteristics of domains that indicate whether the case-based approach will be advantagous and if so, where in the problem solving process it should be useful.

Using the understand, plan, and evaluate model of problem solving developed earlier, different task domain characteristics can be examined with respect to their potential relevance to the case-based approach. In some domains, problems may be hard to understand. For example, algebra word problems (e.g., Hayes and Simon, 1979; Paige and Simon, 1979) appear impossible to novice high school students who lack the ability to reformulate textual problem descriptions into algebraic equations, classify problems (e.g. age problems, volume problems, etc.), or elaborate their problem descriptions with world knowledge (e.g. the sum of the parts of an object can not exceed the original object). In similar problem domains (physics problems, or diagnosis of illness etc), the retrieval of a previous example problem can aid the understanding process. In the sense that a domain can be characterized in terms of difficulty in problem understanding, the case-based approach will be applicable during problem classification or elaboration.

Other problem domains may not be as hard to understand. For example, there is some evidence (Ross, 1982) that novices learning text-editing have no difficulty in understanding the problem (e.g. "insert the new word at the end of the line"). In these domains, suggesting solutions to well understood problems may be the hard part. In these domains, the difficulties associated with complex plans or procedures should yield advantages for the recall of specific cases that can be used to guide the selection and instantiation of a plan in a similar situation. We expect that in these domains, case-based reasoning will be applicable to the planning stage of problem solving.

Case-based reasoning appears most applicable when the following general conditions exist in the problem domain:

1. important domain components vary relatively infrequently
2. problems are presented on a regular basis
3. problems have a certain underlying similarity
4. it takes longer or "costs" more, in general, to compute a new answer than to retrieve, transfer, and modify an old answer.

Case-based reasoning exploits these four characteristics of problems. First, when important components of the domain vary infrequently then decisions made once can be used and reused without needing constant update and recomputing. Second, problems must reoccur with some regularity to warrant the maintenance of the long term memory necessary for case-based reasoning, otherwise, it might be more appropriate to calculate the results from scratch each time. Third, the domain problem should have some underlying similarity in domain concepts so that problems can be related to each other. Finally, the overall effort ("cost") required to maintain and use a long term conceptual memory must be less than what would be required to compute the solutions from scratch. In this sense, case-based reasoning can be viewed as an operational definition for "cognitive economy" (Lenat et al., 1979).

8.1.3 Integrating learning and problem solving

By integrating learning and problem soloving using case-based reasoning, we make more knowledge available to make the problem solving process more efficient. At the same time, the problem solving process provides the important focus of attention necessary to constrain the analogical reasoning process. For example, the task demands of the problem solver during plan selection indicate that the plan type used in a recalled case should be the focus for transfer. This type of focus has been found to be a major requirement in helping a reasoner use analogies effectively (Burstein, 1983; Gentner, 1982).

We claim that case-based reasoning is heuristically more efficient than reasoning from scratch for each problem. At several points in our discussion we have pointed out why we believe this to be so (e.g., sections 4.5.2 or 5.4.4). In order to make the point one last time, we will extract equivalent portions of the reasoning required to make several decisions first using default reasoning and then using case-based reasoning.

First, consider the planning process. Let us assume that we have seven abstract plans at the highest level of abstraction in a planning space (we had seven in the mediation domain so this seems reasonable). Next we assume there are seven general plans for each abstract plan at the next lower level of abstraction. And finally, there are seven specific plans for each general plan at the lowest level of abstraction. This planning space has 343 specific plans from which a planner must select the appropriate plan. In order not to stack the deck too badly, let us assume that the specific plan called PLAN3.3.3 (i.e., the third abstract and

the third general and the third specific plan) is the appropriate one. Using default reasoning, a problem solver would perform the 12 reasoning steps shown in Figure 8-1 in order to select the required specific plan. We use a breadth-first search, but the number of reasoning steps would be the same for a depth-first search.

DEFAULT REASONING STEPS REQUIRED FOR PLAN SELECTION

1. TEST PRECONDITIONS ABSTRACT PLAN1
2. TEST PRECONDITIONS ABSTRACT PLAN2
3. TEST PRECONDITIONS ABSTRACT PLAN3
4. SELECT ABSTRACT PLAN3
5. TEST PRECONDITIONS GENERAL PLAN3.1
6. TEST PRECONDITIONS GENERAL PLAN3.2
7. TEST PRECONDITIONS GENERAL PLAN3.3
8. SELECT GENERAL PLAN3.3
9. TEST PRECONDITIONS SPECIFIC PLAN3.3.1
10. TEST PRECONDITIONS SPECIFIC PLAN3.3.2
11. TEST PRECONDITIONS SPECIFIC PLAN3.3.3
12. SELECT SPECIFIC PLAN3.3.3

Figure 8-1

Now we will look at the equivalent plan selection process using case-based reasoning. For this hypothetical example, we will assume that a case has not already been retrieved from memory during the understanding phase and that the recalled case provides exactly the proper plan (i.e., PLAN3.3.3). The five reasoning steps required are shown in Figure 8-2.

CASE-BASED REASONING STEPS REQUIRED FOR PLAN SELECTION

1. RETRIEVE SIMILAR CASES FROM MEMORY
2. SELECT MOST APPROPRIATE CASE
3. FOCUS ON PLAN USED IN RETRIEVED CASE
4. TEST PRECONDITIONS SPECIFIC PLAN3.3.3
5. SELECT SPECIFIC PLAN3.3.3

Figure 8-2

In a similar fashion, imagine that a problem solver, during failure recovery, were required to incrementally backtrack through each level of reasoning in order to do blame assignment. In section 5.4.3, we described the six levels of reasoning that might have to be investigated in the worst case by default reasoning before a failure could be diagnosed. At each level any number of inferences might have to be checked in order to detect a violation. In a crude way, the levels shown in Figure 8-4 provide an estimate on the number of reasoning steps required in default blame assignment if the failure were caused by an error in problem classification during understanding.

REASONING STEPS REQUIRED IN DEFAULT FAILURE CLASSIFICATION

1. CHECK PREDICTION GENERATION
   ||
   \/
2. CHECK PLAN INSTANTIATION DECISIONS
   ||
   \/
3. CHECK PLAN SELECTION DECISIONS
   ||
   \/
4. CHECK PLANNING POLICY DECISION
   ||
   \/
5. CHECK ELABORATIONAL INFERENCES
   ||
   \/
6. CHECK CONTEXTUAL INFERENCES

Figure 8-3

For case-based reasoning, the same basic five reasoning steps shown in Figure 8-2 (except that the failure class is transferred) would be required to perform the same failure classification task (assuming that there was a similar failure in memory). Using this line of reasoning, we are convinced that case-based reasoning offers the possibility of improving the efficiency of problem solving.

8.2 Some comparisons to other work

In this section we will compare and contrast our research with some related research in artificial intelligence and cognitive science. This comparison will focus on four dimensions:

1. other problem solving models
2. other planning approaches
3. other models of dynamic memory
4. other learning systems

8.2.1 Other problem solving models.

The GPS model of problem solving (Newell and Simon, 1972) is one of the first general heuristic problem solving models in AI and cognitive science. It was based on the heuristic search of a problem space represented as domain specific states which could be altered via application of known operators. Operators were selected via a single line of reasoning known as means-ends analysis with operator subgoaling. In much of AI and cognitive science, the GPS model and other hierarchical planning models (e.g., ABSTRIPS, NOAH, etc.) are equated with models of problem solving. In terms of our case-based model, these are only the planning stage of an integrated problem solver. In our later discussion of planning, we will relate hierarchical planners to other planning approaches.

In many respects, the top level behavior of the MEDIATOR, especially during the time that there are few cases in memory, corresponds roughly to the generate and test behavior exhibited by a GPS problem solver. For example, the MEDIATDR will reason out a solution, fail, recover from failure, reason out a new solution, etc. The details of how these actions are carried out are very different, but the external behaviors are consistent with a type of problem solving behavior described by Newell and Simon (1972) as "creative" problem solving. Because GPS planners cannot learn from their experience, however, they are doomed to remain perpetual novices. The case-based model, on the other hand, provides a mechanism to allow a problem solver to transition from novice behavior based on experience.

The MEDIATOR uses many different types and levels of knowledge to resolve a dispute. GPS always requires specific types of knowledge about goals, states, and operators. For example, GPS cannot infer its goals, nor use knowledge about the problem space and state specific selection decisions (e.g., planning policy criteria for meta-planning). For these reasons even if GPS stored its previous case experiences, it could not reason about the appropriateness of potential transfers from those results. In addition to these issues, the case-based model provides for the recursive use of the model for failure explanation and recovery which were not addresssed within the GPS framework.

The STUDENT model of solving algebra word problems (Bobrow, 1968; Hayes and Simon, 1979; Paige and Simon, 1979) is a model of problem solving that was originally designed to investigate the use of a restricted set of natural language for communication with an automated problem solving system. While not intended as a general model of problem solving, it implicitly included many of the same components that we have made explicit in our problem solving model. The STUDENT system included programs that took an english-like representation of algebra word problems and built an internal propositional representation of them. These propositions were then transformed into equations that were passed to a simple deductive GPS like program that attempted to solve them. In its design, the STUDENT model implicitly included separate understanding, planning, and executing phases. It even included a simple form of failure recovery by substituting alternate interpretations of variables when it failed to find a solution.

During the understanding process, STUDENT's approach to building its representation was basically a syntactic process of translation. The MEDIATOR, on the other hand, has a knowledge intensive approach to problem understanding that uses domain knowledge to recognize conflicts that arise as the problem representation evolves. Even though STUDENT had special routines to handle "age problems," there was no recognition of the fact that algebra word problems could be grouped into classes and that these classes could be used to organize solution methods. Subsequent research has shown this to be an important aspect to human problem solving performance (Hinsley et al., 1977; Chi et al., 1981) and has been explicitly included in the case-based model of problem solving. The other important difference, of course, includes the fact that STUDENT could learn only by being told. There was no capability to record the program's experience so that it could be used in later problem solving. STUDENT, therefore, would solve the same problem twice by repeating the same long line of reasoning.

The blackboard architecture is a problem solving model originally developed for the Hearsay-II speech-understanding system (Erman et al., 1980). In the years since its use in Hearsay-II, the informal use of the blackboard architecture has proven remarkably versatile in a wide range of AI systems. It has been employed for vehicle tracking and planning, sonar signal interpretation, multiple-task planning, protein crystallography, and scene analysis.

The blackboard framework has also been used in psychological investigations of the reading process, text comprehension, and composition planning (Hayes-Roth, 1983). Adding our use of the generic blackboard structure as a framework for case-based problem solving outside the usual rule-based paradigm to these uses lends further evidence to the generality of this architecture.

In adopting the blackboard architecture, we freely modified the components as necessary to match our research needs. This has been the case with all blackboard implementations to date. Despite this tendancy of various researchers to modify the pieces, the basic elements of a blackboad architecture appear reasonably consistent:

1. intermediate results held in a working memory
2. independent knowledge sources which can change working memory
3. a structured global data base that provides additional knowledge and organizes previous results, and
4. an intelligent control mechanism that decides when and how different knowledge sources will be employed.

In the MEDIATOR, the working memory element of the blackboard corresponds to the "case frame" that is constructed during problem solving. The frame retains all the intermediate results of problem understanding and planning until the problem has been successfully resolved. The independent knowledge sources element of the blackboard architecture corresponds to our different components of mediation cases. For example, the MEDIATOR's semantic knowledge of disputants, dispute arguments, disputed objects, goals, plans, etc. These knowledge sources in combination with default reasoning provided one set of possible changes to working memory. The global data base of the general blackboard architecture is, of course, represented by our conceptual memory of cases. This supplies previous results that are the basis for case-based reasoning and its heuristic changes to working memory. Finally, the control mechanism of the blackboard is represented in the MEDIATOR as the process model of problem solving that specifies the default sequence of understand, plan, and follow-up of failure decisions.

Some of the differences between our implementation and other blackboard models include the fact that we integrate the processes involved in problem solving, (i.e., the knowledge and rationale for decisions) in addition to the end results of reasoning from multiple knowledge sources. The use of multiple sources in blackboards has been suggested as a heuristic that avoids failures. We have gone further by showing how the blackboard architecture can be employed to accomplish blame assignment and failure recovery when failures do occur. In addition, no previous blackboard implementation has attempted to use a global data base modeled after human memory organization and retrieval processes as has been demonstrated in this research. Because we have explicitly included an episodic memory as a global data base, we also have been able to demonstrate learning in a blackboard architecture for the first time.

8.2.2 Other AI planning approaches

Our basic (i.e., without case-based reasoning) approach to planning is best classified as a "plan instantiation" approach. This approach can be contrasted to three others recognized in AI research. These are the nonhierarchical, hierarchical, and opportunistic planning approaches (Cohen and Feigenbaum, 1982). These approaches, while different in important ways, are not mutually exclusive. For example, the key difference between the hierarchical, plan instantiation, and opportunistic approaches versus nonhierarchical planners is that the former "represent" plans on several levels of abstraction, while the latter have only one level of plan representation. This makes nonhierarchical planning systems (e.g., Fikes, et al., 1972) much less efficient since unprofitable planning alternatives are often pursued and much detailed planning wasted. This problem is avoided by the other planning approaches because they pursue more detailed planning only when an abstract solution is believed to solve the problem.

The opportunistic approach to planning (Hayes-Roth and Hayes-Roth, 1979) can be differentiated from the other approaches in two ways. First, it is the only one of the approaches that has been advanced as a cognitive model of human planning. Second, its flexible control strategy using the "blackboard" control structure (Erman, et al., 1980) allows both bottom-up planning (i.e., it watches for "opportunities" to make detailed planning decisions) as well as the top-down refinement method of the hierarchical and plan instantiation approaches. Even though hierarchical and opportunistic planners are similar in many ways, they can be differentiated in terms of the amount of structure that exists in the domain and the level of experience of the planner (Hayes-Roth and Hayes-Roth, 1979). This difference is suggested by the diagram shown in Figure 8-4.

Comparison of Planning Approaches by Domain and Expertise

	Domain Characteristics	Planner Expertise
Hierarchical Planning	Structured Domain Familiar problems	Practiced planner Well-learned plans
Opportunistic Planning	No Structure Novel problems	Inexperienced planner

Figure 8-4

We have chosen not to adopt the opportunistic approach to planning for two reasons. First, we feel that problem solvers impose structure on problem domains in order to reason effectively in the face of resource limitations. This view is supported by empirical investigations that were discussed in section 8.1.1. Second, we are interested in the transition of a problem solver from novice to expert. Despite the appearance of opportunistic behavior on the part of novice problem solvers, we can explain the same behavior in our model by having many iterations of planning and internal failure recovery. Nonetheless, we have found the concept of a global control structure, such as the "blackboard," to be an important control consideration in our model of case-based problem solving as discussed in chapter five.

In both opportunistic and hierarchical planning systems (e.g., Sacerdoti, 1977; Stefik, 1981; Wilkins, 1984), plans are constructed from scratch for each problem. This contrasts with the plan instantiation approach where plans are selected from a set of already known abstract plan types. In plan instantiation systems (e.g., Friedland, 1979; Wilensky, 1983), the most promising general plan is selected at the highest level of abstraction and then successively refined until fully instantiated. We have adopted this approach as our default planning methodology because it is more compatibile with our research objectives of avoiding the computation required to recreate a plan from scratch for two similar problems.

Planning in the face of multiple goals, for most hierarchical planners (Sacerdoti, 1977; Wilensky, 1983; Wilkins, 1984), involves constructing a plan and then criticizing it for negative goal interactions. Our case-based model of planning explicitly includes the knowledge of goal interactions so that they can be used as part of the plan selection process. This allows the MEDIATOR, for example, to avoid having to go through this type of internal evaluation, failure and backtracking.

Our use of case-based reasoning to augment the plan instantiation approach has served to confirm many of the observations of Carbonell (1983, 1983b) on analogical problem solving and derivational analogy. He chose to combine his analogical reasoning processes with a "means-ends analysis" hierarchical planner (Newell and Simon, 1972). When taken together with Carbonell's work, there seems to be sufficient evidence to prescribe case-based or other forms of analogical reasoning as applicable heuristics to support any planning approach. Despite the similarities to Carbonell's (1983) work in analogical problem solving, there are several differences. First, he only used the weak MEA method of planning and didn't investigate more powerful domain specific planning such as plan instantiation. Second, although he discusses the use of a MOP-like memory for analogical problem solving, he never specified the update and retrieval details necessary to effect an integration of a problem solving model with a dynamic long term memory. Finally, he did not specify any way of focusing analogical transfer in the problem solving process. This, in general, is a hard problem for previous models of reasoning by analogy (e.g., Burstein, 1983; Gentner, 1982; Winston, 1975, 1980). Because we make the stages of problem solving explicit in our model, we can use this structure of the process to focus on the portions of the analogy that are considered for transfer.

Another computer program that performs analogical reasoning was developed by Evans (1968). His program, called ANALOGY, could solve geometric analogy problems such as those used on standard intelligence tests. As in the MEDIATOR, ANALOGY used a weighted scoring function to choose between competing analogies. In Evan's scheme, he biases the analogy selection toward analogies involving rotations and against reflections (without apparent appeal to cognitive plausibility). Salient features of his geometric domain include reflection, rotation, scale, and combinations, as well as no change.

8.2.3 Other models of memory.

Previous work in "conceptual memory" (Schank, 1980, 1982; Lebowitz, 1980; Kolodner, 1984) has provided insight into how an organized knowledge base of experiential information can be built up and accessed. This long term memory model of experience is the framework on which this research was built. The problems of organizing and retrieving events in a long term memory were initially explored in the program CYRUS (Kolodner, 1984). CYRUS was an intelligent fact retrieval system whose episodic memory was composed of conceptual categories that collectively partitioned the range of experience expected in the everyday life of a Secretary of State. These conceptual categories, called E-MOPs (Episodic Memory Organization Packets), provided for reconstructive access to the events within each category based on

traversal of conceptually discriminative indices. Our generalized episodes are closest to Kolodner's implementation of E-MOP's. Part of the motivation for the reconstructive approach was due to the psychological evidence showing that people seem to employ reconstruction in their recall of experiences (Bartlett, 1932; Spiro et al., 1978; Loftus, 1979; Neisser, 1981; Williams and Hallan, 1981; Reiser and Black, 1983). The key features of this organization are (Kolodner, 1984, 1981):

(1) Each node is composed of both generalized normatives abstracted from experiences as well as pointers to more specialized details of those events and other related conceptual categories.

(2) More specialized experiences are indexed by their variance from the generalized norms.

(3) Index traversal requires both identification of a feature type and specific feature value.

(4) Indices within the conceptual categories provide important discriminations.

Within the context of the above organization, maintenance and reconstructive retrieval of experiences were demonstrated using a series of task specific strategies. Thus memory maintenance strategies demonstrated the reorganization of an evolving knowledge base by employing generalization and specialization reasoning to integrate new events into memory. In this way, experiences which exactly match previously generalized knowledge effectively added no information or burden on memory, while the novel features of experiences were isolated and retained.

Question answering in CYRUS is initiated when a question, which contains a target concept, is presented. The question concept is classified into a question category and then elaborated to determine the questions' "intent." Once the question has been understood, the target concept is used to drive the sequential selection of conceptual categories, the selection of indices that must be traversed, and the location of the appropriate place in the memory structure. If an event is found in memory at the location the target concept would normally be indexed, then that event is available for use in generating the appropriate response if it matches the target concept.

CYRUS's question answering process is very similar to our problem solving process, except that we obviously generate a solution plan as opposed to an answer to a query. The implementation details for both CYRUS and MEDIATOR dynamic memory processes reflect the differences in their corresponding tasks. The most important differences between their dynamic memories have to do with the nature of the retrieval cue and the restrictions placed on the retrieved cases. The retrieval cue for CYRUS is a question. In the MEDIATOR, the retrieval cue is the problem description. In CYRUS, a single event is returned only if it matches the target concept. In the MEDIATOR, the best "near-miss" case is retrieved for each component of the problem description (both domain problems and failure problems) so that many cases are recalled for each problem presented.

8.2.4 Semantic and episodic memory distinctions

In psychology, there is some debate over the difference between "semantic" and "episodic" memory (Tulving, 1972). While the precise distinctions are easily debatable (e.g., Kolodner, 1984), the development of the MEDIATOR program has provided a nice operational definition of their difference. The program is provided a "semantic" model of its environment in terms of primitive and generalized concept types. These types comprise the "instance" and "generalization" language discussed in chapter six. In our object-oriented implementation, these types are in terms of "Flavor" definitions (Weinreb and Moon, 1981). This is easily equivalent to the conventional notion of semantic memory since these definitions are used at runtime to build a hierarchical network that relates the concept types using the usual "isa" relationship. This is a priori static knowledge provided at runtime and is not modified during the session. This knowledge allows the program to determine the similarity relationship between arbitrary concepts as required by analogical reasoning processes.

The program's "semantic" knowledge is used to guide the instantiation of concepts when the case data files are loaded at program runtime. This process is operationally equivalent to building a "working memory" for the program. As each case is presented to the program, its initial representation in working memory is copied into the local "memory" of the problem solving processes. The actual representation at any stage of processing is dependent on this local "memory" associated with individual subprocess (e.g., classification, plan selection, etc.). This provides an operational definition for the structure psychologists call "short term memory." When the program is finished processing cases, it updates a dynamic data structure that represents a natural parallel to an "episodic" memory. The program's episodic memory, unlike its semantic memory, is idiosyncratic since it depends on the sequence of presented cases and the interaction with the environment at runtime. Using the distinction between semantic and episodic memory provided by the MEDIATOR program, it would be interesting to investigate another process that could inspect a problem solver's episodic memory to create new semantic concepts. The integration of these new concepts into semantic memory would not be difficult with the current implementation. The difficulty would arise in maintaining consistency between old and new concept instances that are affected by having a dynamic semantic

memory.

8.2.5 Other learning systems

The MEDIATOR exhibits three types of learning:

1. rote learning
2. inductive learning
3. learning by taking advice.

All learning in the MEDIATOR happens as a consequence of the integration of cases into the program's episodic long term memory. In this way, it performs "rote learning" of specific case experiences and makes generalizations based on these case instances. In this way, the MEDIATOR is similar to earlier dynamic memory systems: CYRUS (Kolodner, 1984) and IPP (Lebowitz, 1980).

Because memory contains the component features associated with both successful and unsuccessful plan applications, the MEDIATOR can inductively learn the domain of applicability for its plans. This plus its use of the candidate elimination algoritm make the MEDIATOR similar to LEX (Mitchell et al., 1983). One difference between LEX and the MEDIATOR is in blame assignment. LEX generates all possible explanations for failure as alternate rule hypotheses and depends on the problem generator to eliminate the incorrect explanations. The MEDIATOR attempts blame assignment in order to avoid generating all possible explanations. This is especially important because the MEDIATOR has not control over problem presentation order or evaluation by the environment.

The last type of learning performed by the MEDIATOR is learning by being told. This happens in the context of requesting and getting feedback from the environment. Because the feedback may be at a high level, operalization (Mostow, 1983) may be required to make effective use of the advice. This is a very different type of learning than that performed by "knowledge engineering" processes (Davis and Lenat, 1980).

The case-based problem solving process, when viewed from the perspective of a learning system is reflected in Figure 8-5.

CASE-BASED PROBLEM SOLVING VIEWED FROM A LEARNING PERSPECTIVE

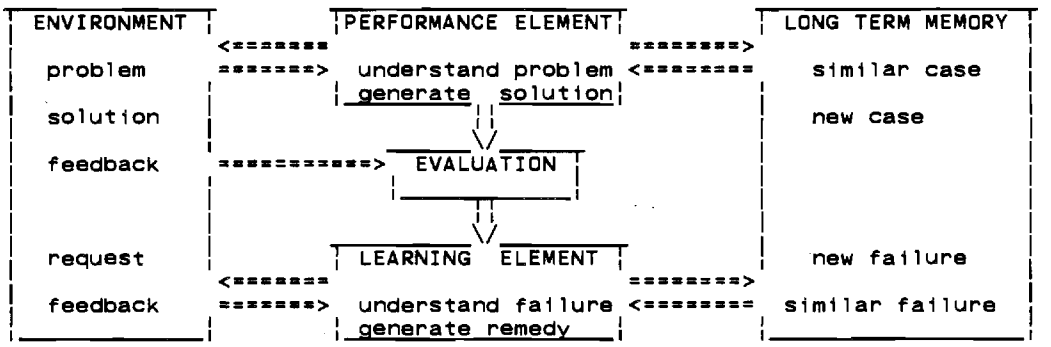


Figure 8-5

8.3 Problem solving paradigms

Winston (1984) has described AI as being primarily concerned with the use of an "armamentarium of problem solving paradigms." He lists the following AI paradigms:

AI Problem Solving Paradigms

1. *Describe-and-match* paradigm - a source problem is described in terms of a domain specific set of features and relationships. This description is then matched against a target description to determine the relationship between the source and the target.

2. *Problem-reduction* paradigm - a complex task is achieved by reducing it into a set of subtasks that can in turn be reduced, etc.

3. *Constraint propagation* paradigm - infer plausible values for a set of variables by propagating to other variables only those alternatives that are locally consistent with some domain specific constraints. After some number of iterations, these local constrains will allow the development of at least one globally consistent interpretation.

4. *Search* paradigm - a space of alternatives is methodically investigated by repeatedly exploring states in the space according to an overall goal directing strategy.

5. *Means-ends analysis* paradigm - a procedure is selected from a set of known procedures according to its ability to reduce a known difference between the current state of a problem and the goal state.

6. *Generate-and-test* paradigm - two basic modules are used: a generator produces possible solutions and a tester evaluates each proposal for acceptance or rejection.

7. *Rule-based* paradigm - all inference is represented in terms of rules of the form "IF <condition> THEN <action>."

8. *Theorem proving* paradigm - using traditional logical notions of predicate calculus and rules of inference, expressions are resolved such that problems are either proved or disproved in much the same fashion as a mathematical proof.

Figure 8-6

On the basis of our research, we propose the following additional paradigm to the AI armamentarium:

CASE-BASED REASONING PROBLEM SOLVING PARADIGM

Before attempting a long static decision making process, try to remember a previous similar case to see if a similar decision might be transferred and applied here as well.

Figure 8-7

REFERENCES

Anderson, J. R. Cognitive Psychology and Its Implications. San Francisco: W. H. Freeman and Co., 1980.

Anderson, J. R., Boyle, C. F., Farrell, R., and Reiser, B. Cognitive Principles in the design of computer tutors. In Proceedings of the Sixth Annual Conference of the Cognitive Science Society, Boulder, CO., 2-9, 1984.

Aho, A. V., Hopcroft, J. E., and Ullman, J. D. The design and analysis of computer algorithms. Reading, Mass.: Addison-Wesley, 1974.

Bartlett, R. Remembering: A Study in Experimental and Social Psychology. London: Cambridge University Press, 1932.

Barr, A. and Feigenbaum, E. A. (Eds.) The Handbook of Artificial Intelligence, Volumes I and II. Los Altos, CA: William Kaufmann, 1981.

Blair, D. C. and Maron, M. E. An evaluation of retrieval effectiveness for a full-text document-retrieval system. Communications of the ACM, 1985, 28, 289-299.

Bobrow, D. G. Natural language input for a computer problem-solving system. In Minsky, M. (Ed.) Semantic Information Processing. Cambridge, MA: The MIT Press, 1968.

Bransford, J. D., Barclay, J. R., and Franks, J. J. Sentence memory: A constructive versus interpretive approach. Cognitive Psychology, 1972, 3, 428-436.

Buchanan, B. G., and Feigenbaum, E. A. DENDRAL and Meta-DENDRAL. Artificial Intelligence, 1978, 11, 5-24.

Burstein, M. H. A model of learning by analogical reasoning and debugging. In the proceedings of the National Conference on Artificial Intelligence, 22-26 August 1983, in Washington, D. C., 45-48.

Carbonell, J. G. Jr. Subjective Understanding: Computer Models of Belief Systems. Yale University Department of Computer Science Research Report #150, 1979.

Carbonell, J. G. Jr. Metaphor: An inescapable phenomenon in natural-language comprehension. In W. G. Lehnert and M. H. Ringle (Eds.) Strategies for Natural Language Processing, Hillsdale, NJ: Lawrence Earlbaum Associates, 1982.

Carbonell, J. G. Jr. Learning by analogy: formulating and generalizing plans from past experience. In Michalski, R.S., Carbonell, J. G., and Mitchell, T. M. (Eds.) Machine Learning: An Artificial Intelligence Approach. Palo Alto: Tioga Publishing Co., 1983.

Carbonell, J. G. Jr. Derivational analogy and its role in problem solving. and In the Proceedings of the National Conference on Artificial Intelligence, 22-26 August 1983a, in Washington, DC, 64-69.

Carbonell, J. G. Jr. Derivational analogy in problem solving and knowledge acquisition. In the Proceedings of the International Machine Learning Workshop, 22-24 June 1983b, in Monticello, Illinois, 12-18.

Chandrasekaran, B. Towards a taxonomy of problem solving types. The AI Magazine, Winter-Spring 1983, 4, 9-17.

Charniak, E. A common representation for problem solving and language comprehension information. Artificial Intelligence, 1983, 16.

Charniak, E., Reisbeck, C. K., and McDermott, D. V. Artificial Intelligence Programming. Hillsdale, NJ: Lawrence Earlbaum Associates, 1980.

Chase, W. G. and Simon, H. A. Perception in chess. Cognitive Psychology, 1973, 4, 55-81.

Chi, M. T. H.; Feltovich, P. J. and Glaser, R. Categorization and representation of physics problems by experts and novices. Cognitive Science, 1981, 5, 121-152.

Clement, J. Analogy generation in scientific problem solving. Proceedings of the Third meeting of the Cognitive Science Society, 1981, 137-140.

Clement, J. Analogical reasoning patterns in expert problem solving. Proceedings of the Fourth Annual Conference of the Cognitive Science Society, 1982.

Cohen, P. R. and Feigenbaum, E. A. (Eds.) The Handbook of Artificial Intelligence, Volume III. Los Altos, CA: William Kaufmann, 1982.

Cullingford, R. SAM. In Schank, R. C. and Riesbeck, C. K. (Eds.) Inside Computer Understanding. Hillsdale, NJ: Lawrence Earlbaum Associates, 1981.

Davis, R. and Lenat, D. B. Knowledge Based Systems in Artificial Intelligence. New York: McGraw-Hill, 1980.

de Kleer, J., Doyle, J., Steele, G. L., and Sussman, G. J. Explicit control of reasoning. In P. H. Winston and R. H. Brown (Eds.) Artificial Intelligence: An MIT Perspective. Cambridge MA.: The MIT Press, 1979.

Dietterich, T. G. and Michalski, R. S. A comparative review of selected methods for learning from examples. In Michalski, R.S., Carbonell, J. G., and Mitchell, T. M. (Eds.) Machine Learning: An Artificial Intelligence Approach. Palo Alto: Tioga Publishing Co., 1983.

Doyle, J. A glimpse of truth maintenance. In P. H. Winston and R. H. Brown (Eds.) Artificial Intelligence: An MIT Perspective. Cambridge MA.: The MIT Press, 1979.

Dyer, M. G. In-Depth Understanding: A Computer Model of Integrated Processing for Narrative Comprehension. Cambridge, MA.: The MIT Press, 1983.

Erman, L. D., Hayes-Roth, F., Lesser, V. R., and Reddy, D. R. The Hearsay-II speech-understanding system: integrating knowledge to resolve uncertainty. Computing Serveys, 1980, 12, 213-253.

Evans, T. G. A program for the solution of geometric-anology intelligence test questions. In Minsky, M. (Ed.) Semantic Information Processing. Cambridge, MA: The MIT Press, 1968.

Fikes, R. E., Hart, P. E., and Nilsson, N. J. Learning and executing generalized robot plans. Artificial Intelligence, 1972, 3, 251-288.

Fisher, R. and Ury, W. with Bruce Potter (Ed.). Getting to Yes. Boston: Houghton Mifflin Co., 1981.

Flowers, M. On being contradictory. In the proceedings of the National Conference on Artificial Intelligence, 18-20 August 1982, in Pittsburgh, PA., 269-272.

Flowers, M., McGuire, R., and Birnbaum, L. Adversary arguments and the logic of personal attack. In W. G. Lehnert and M. H. Ringle (Eds.) Strategies for Natural Language Processing, Hillsdale, NJ: Lawrence Earlbaum Associates, 1982.

Franks, J. J. and Bransford, J. D. Abstraction of visual patterns. Journal of Experimental Psychology, 1971, 90, 65-74.

Friedland, P. E. Knowledge-based experiment design in molecular genetics. Rep. No. 79-771, Computer Science Dept., Stanford University, 1979. Referenced in Cohen, P. R. and Feigenbaum, E. A. (Eds.) The Handbook of Artificial Intelligence, Los Altos, CA: William Kaufmann, Inc.

Gentner, D. Structure Mapping: a theoretical framework for analogy and similarity. In the proceedings of the Fourth Annual Conference of the Cognitive Science Society, August, 1982, 13-15.

Gershman, A. V., Johnson, P., Shwartz, S. and Wolf, T. CD meets the real world. In the proceedings of the First Annual Workshop on Theoretical Issues in Conceptual Information Processing, March, 1984, 61-80.

Gick, M.L. and Holyoak, K.J. Analogical problem solving. Cognitive Psychology, 1980, 12, 306-355.

Gick, M.L. and Holyoak, K.J. Schema induction and analogical transfer. Cognitive Psychology, 1983, 15, 1-38.

Granger, R. H.; Eiselt, K. P.; and Holbrook, J. K. The parallel organization of lexical, syntactic, and pragmatic inference processes. In the proceedings of the First Annual Workshop on Theoretical Issues in Conceptual Information Processing, March, 1984, 97-106.

Green, D. M. and Swets, J. A. (1966) Signal Detection Theory described and referenced in Wingfield, A. and Byrnes, D. L. The Psychology of Human Memory. New York: Academic Press, 1981.

Greeno, J. G. "Process of understanding in problem solving." In Castellan, N. J., Pisoni, D. B., and Potts, G. R. (Eds.) Cognitive Theory Volume 2. Hillsdale, NJ.: Lawrence Erlbaum, 1977.

Haefner, M. E. An advisory system for psychiatric diagnosis. In the proceedings of the First Annual Workshop on Theoretical Issues in Conceptual Information Processing, March, 1984, 117-127.

Hammond, K. J. Planning and Goal Interaction: the use of past solutions in present situations. In the proceedings of the National Conference on Artificial Intelligence, 22-26 August 1983, in Washington, D. C., 148-151.

Hart, P. E., Nilsson, N. J., and Raphael, B. A formal basis for the heuristic determination of minimum cost paths, IEEE Trasactions on Systems Science and Cybernetics, 1968, 100-107.

Hayes, J. A. The Complete Problem Solver. Philadelphia, PA: The Franklin Institute Press, 1981.

Hayes, J. A. and Simon, H. A. Understanding written problem instructions. In H. A. Simon Models of Thought, New Haven: Yale University Press, 1979.

Hayes-Roth, B. The blackboard architecture: a general framework for problem solving? Heuristic Programming Project Report No. HPP-83-30, Computer Science Department, Stanford University, May, 1983.

Hayes-Roth, B. and Hayes-Roth, F. Concept learning and the recognition and classification of exemplars. Journal of Verbal Learning and Verbal Behavior, 1977, 16, 321-338.

Hayes-Roth, B. and Hayes-Roth, F. A cognitive model of planning. Cognitive Science, 1979, 3, 275-310.

Hayes-Roth, F. Using proofs and refutations to learn from experience. In Michalski, R.S., Carbonell, J. G., and Mitchell, T. M. (Eds.) Machine Learning: An Artificial Intelligence Approach. Palo Alto: Tioga Publishing Co., 1983.

Hayes-Roth, F., Waterman, D. A., and Lenat, D. B. (Eds.) Building Expert Systems. Reading, MA: Addison-Wesley, 1983.

Hinsley, D. A., Hayes, J. R., and Simon, H. A. From words to equations: meaning and representation in algebra word problems. In M. A. Just and P. A. Carpenter (Eds.) Cognitive Processes in Comprehension. Hillsdale, N.J.: Lawrence Erlbaum Associates, 1977.

Kolodner, J. L. On Beyond CYRUS: Additional Problems Related to Long Term Memory Organization and Retrieval. Technical Report GIT-ICS-81/17, Georgia Institute of Technology, School of Information and Computer Science, Atlanta, GA., 1981.

Kolodner, J. L. Towards an Understanding of the Role of Experience in the Evolution from Novice to Expert. International Journal of Man-Machine Studies, 1983, 19, 497-518.

Kolodner, J. L. Retrieval and Organizational Strategies in Conceptual Memory: A Computer Model. Hillsdale, NJ: Lawrence Erlbaum Associates, 1984.

Lebowitz, M. Generalization and Memory in an Integrated Understanding System. Research Report #186, Yale University, Department of Computer Science, New Haven, CT., 1980.

Lehnert, W. G. The Process of Question Answering. Hillsdale, NJ: Lawrence Earlbaum, 1978.

Lenat, D. B., Hayes-Roth, F., and Klahr, P. Cognitive economy in artificial intelligence systems. In the Proceeding of the Sixth International Joint Conference on Artificial Intelligence, Tokoyo, Japan, 531-536, 1979.

Lichtenstein, E. H. and Brewer, W. F. Memory for goal-directed events. Cognitive Psychology, 1980, 12, 412-445.

Loftus, E. F. Leading questions and the eyewitness report. Cognitive Psychology. 1975, 7, 560-572.

Loftus, E. F. Reconstructive memory processes in eyewitness testimony. In Perspectives in law and psychology, ed. B. D. Sales. New York: Plenum, 1978.

Loftus, E. F. Eyewitness Testimony. Cambridge, Massachusetts: Harvard University Press, 1979.

Loftus, E. F. and Zanni, G. Eyewitness testimony: the influence of the wording of a question. Bulletin of the Psychonomic Society, 1975, 5, 86-88.

Luchins, A. S. Mechanization in problem solving. Psychological Monographs, 1942, 54, no. 248.

Lukov, V. B., Sergeev, V. M., and Tyulin, I. G. Reflective model of negotiations process. IEEE Technology and Society Magazine, June 1984, 20-27.

Main, J. How to be a better negotiator. FORTUNE, 19 September 1983, 141-146.

May, H. G. and Metzger, B. M. (Eds.) The Oxford Annotated Bible with the Apocrypha, Revised Standard Edition. New York: Oxford University Press, 1965, page 419.

Mayer, R., Larkin, J., and Kadane, J. A cognitive analysis of mathematical problem-solving ability. In R. Sternberg (Ed.) Advances in the Psychology of Human Intelligence, 1984.

McCarthy, John (1958). Programs with common sense. In Minsky, M. (Ed.) Semantic Information Processing. Cambridge, MA: The MIT Press, 1968.

Medin, D. L. and Schaffer, M. M. Context theory of classification learning. Psychological Review, 1978, 85, 207-238.

Minsky, M. L. Steps toward artificial intelligence. In Feigenbaum, E. A. and Feldman, J. (Eds.). Computers and Thought New York: McGraw-Hill, 1963.

Minsky, M. L. Matter, mind, and models. In Minsky, M. (Ed.) Semantic Information Processing. Cambridge, MA: The MIT Press, 1968.

Minsky, M. L. A framework for representing knowledge. In P. H. Winston (Ed.) The Psychology of Computer Vision. New York: McGraw-Hill, 1975.

Mitchell, T. M. Generalization as search. Artificial Intelligence, 1981, 18, 203-226.

Mitchell, T. M., Utgoff, P. E., and Banerji, R. Learning by experimentation: Acquiring and refining problem solving heuristics. In Michalski, R.S., Carbonell, J. G., and Mitchell, T. M. (Eds.) Machine Learning: An Artificial Intelligence Approach. Palo Alto: Tioga Publishing Co., 1983.

Mostow, D. J. Machine transformation of advice into a heuristic search procedure. In Michalski, R.S., Carbonell, J. G., and Mitchell, T. M. (Eds.) Machine Learning: An Artificial Intelligence Approach. Palo Alto: Tioga Publishing Co., 1983.

Neisser, U. John Dean's memory: a case study. Cognition, 1981, 9, 1-22.

Newell, A. and Simon, H. Human Problem Solving. Englewood Cliffs, NJ.: Prentice-Hall, 1972.

Norman, D. A., Gentner, D. R., and Stevens, A. L. (1976) Description of analogy in the "Mayonnaise Problem" referenced in Wingfield, A. and Byrnes, D. L. The Psychology of Human Memory. New York: Academic Press, 1981.

O'Rorke, P. Reasons for beliefs in understanding: applications of non-monotonic dependencies to story processing. In the proceedings of the National Conference on Artificial Intelligence, 22-26 August 1983, in Washington, D. C., 306-309.

Paige, J. M. and Simon, H. A. Cognitive processes in solving algebra word problems. In H. A. Simon Models of Thought, New Haven: Yale University Press, 1979.

Polya, G. How To Solve It. Princeton, NJ.: Princeton University Press, 1945.

Quillian, M. R. Semantic Memory. In Minsky, M. (Ed.) Semantic Information Processing. Cambridge, MA: The MIT Press, 1968.

Raiffa, H. The Art and Science of Negotiation. Cambridge, Mass: Havard University Press, 1982.

Reed, S. K. Pattern recognition and categorization. Cognitive Psychology, 1972, 3, 382,407.

Reed, S. K.; Ernst, G.W. and Banerji, R. The role of Analogy in transfer between similar problem states. Cognitive Psychology, 1974, 6, 436-450.

Reed, S. K. and Johnsen, J. A. Memory for problem solutions. In Bower, G. H. (Ed.) The Psychology of Learning and Motivation. New York: Academic Press, 1977.

Reiser, B. J. and Black, J. B. The roles of interference and inference in the retrieval of autobiographical memories. In the proceedings of the Fifth Annual Conference of the Cognitive Science Society, 18-20 May 1983, University of Rochester, Rochester, New York.

Rifkin, A. A model of knowledge representation based on deontic modal logic. In the proceedings of the Sixth Annual Conference of the Cognitive Science Society, 28-30 June 1984, University of Colorado, Boulder, Colorado.

Rissland, E. Examples in the legal domain: hypotheticals in contract law. In the proceedings of the Fourth Annual Conference of the Cognitive Science Society, 1982, Ann Arbor, Michigan.

Rissland, E. Learning how to argue: using hypotheticals. In the proceedings of the First Annual Workshop on Theoretical Issues in Conceptual Information Processing, 1984, Atlanta, Georgia.

Rosch, E. Classification of real-world objects: origins and representations in cognition. In Johnson-Laird, P. N. and Wason, P. C. (Eds.) Thinking, Readings in Cognitive Science. Cambridge: Cambridge University Press, 1977.

Ross, Brian H. Remindings and Their Effects in Learning a Cognitive Skill. Cognitive and Instructional Sciences Series CIS-19. Palo Alto, CA: Xerox Palo Alto Research Centers, 1982.

Rouse, W. B. Models of human problem solving: detection, diagnosis, and compensation for system failures. Automatica, 1983, 19, 613-625.

Rumelhart, D. E. and Abrahamson, A. A. A model for analogical reasoning. Cognitive Psychology, 1973, 5, 1-28.

Rychener, M. The instructable production system: a retrospective analysis. In Michalski, R.S., Carbonell, J. G., and Mitchell, T. M. (Eds.) Machine Learning: An Artificial Intelligence Approach. Palo Alto: Tioga Publishing Co., 1983.

Sacerdoti, E. D. A Structure for Plans and Behavior. Amsterdam: Elsevier North-Holland, 1977.

Salzberg, S. Generating hypotheses to explain prediction failures. In the proceedings of the National Conference on Artificial Intelligence, 22-26 August 1983, in Washington, D. C., 352-355.

Samuel, A. L. Some studies in machine learning using the game of checkers. In Feigenbaum, E. A. and Feldman, J. (Eds.). Computers and Thought New York: McGraw-Hill, 1963.

Schank, R. C. Conceptual dependency: a theory of natural language understanding. Cognitive Psychology, 1972, 3, 552-631.

Schank, R. C. Language and memory. In D. A. Norman (Ed.) Perspectives on Cognitive Science. Norwood NJ: Ablex Publishing Company, 1980.

Schank, R. C. Dynamic Memory. Cambridge: Cambridge University Press, 1982.

Schank, R. C. The current state of AI: one man's opinion. The AI Magazine, Winter-Spring 1983, 4, 3-8.

Schank, R. C. and Abelson, R. P. Scripts, Plans, Goals, and Understanding. Hillsdale, NJ: Lawrence Earlbaum Associates, 1977.

Schank, R. C. and Birnbaum, L. Memory, Meaning, and Syntax. Research Report #189, Yale University, Department of Computer Science, New Haven, CT., 1980.

Schank, R. C. and Carbonell, J. G. Jr. Re: The Gettysburg Address, Representing Social and Political Acts. Research Report #127, Yale University, Department of Computer Science, New Haven, CT., 1978.

Schank, R. C. and Riesbeck, C. K. Inside Computer Understanding. Hillsdale, NJ: Lawrence Earlbaum Associates, 1981.

Shortliffe, E. and Buchanan, B. A model of inexact reasoning in medicine. Mathematical Biosciences, 1975, 23, 351-379.

Simkin, W. E. Mediation and the Dynamics of Collective Bargaining. Washington, DC.: The Bureau of National Affairs, Inc., 1971.

Simon, H. A. A behavioral model of rational choice. In H. A. Simon Models of Thought, New Haven: Yale University Press, 1979.

Simon, H. A. Why should machines learn? In Michalski, R.S., Carbonell, J. G., and Mitchell, T. M. (Eds.) Machine Learning: An Artificial Intelligence Approach. Palo Alto: Tioga Publishing Co., 1983.

Smith, E. E.; Adams, N.; and Schorr, D. Fact retrieval and the paradox of interference. Cognitive Psychology, 1978, 10, 438-464.

Smith, E. E. and Medin, D. L. Categories and Concepts. Cambridge, MA: Havard University Press, 1981.

Spiro, R. J., Esposito, J., and Vondruska, R. J. The representation of derivable information in memory: When what might have been left unsaid is said. In TINLAP-2: Theoretical Issues in Natural Language Processing, Sponsored by the Association for Computing Machinery. University of Illinois, 1978, 226-231.

Stallman, R. M. and Sussman, G. J. Problem solving about electric circuits. In P. H. Winston and R. H. Brown (Eds.) Artificial Intelligence: An MIT Perspective. Cambridge MA.: The MIT Press, 1979.

Stefik, M. Planning and meta-planning (MDLGEN: part 2). Artificial Intelligence, 1981, 16, 141-170.

Sterling, T. D. Unionization of professionals in data processing: an assessment of recent trends. Communications of the ACM, 1982, 25, 807-816.

Sternberg, R. J. Component processes in analogical reasoning. Psychological Review, 1977, 84, 353-378.

Sulin, R. A. and Dooling, D. J. Intrusion of a thematic idea in retention of prose. Journal of Experimental Psychology, 1974, 103, 255-262.

Sussman, G. J. A Computer Model of Skill Acquisition. New York: American Elsevier Publishing Company, 1975.

Sycara-Cyranski, K. Arguments of persuasion in labour mediation. In the Proceedings of the Ninth International Joint Conference on Artificial Intelligence, 1985.

Tedeschi, J. T. and Rosenfeld, P. Communication in bargaining and negotiation. In Roloff, M. E. and Miller, G. R. (Eds.) Persuasion: New Directions in Theory and Research. Beverly Hills, CA: Sage Publications, 1980.

Tulving, E. Episodic and semantic memory. In E. Tulving and Donaldson (Eds.), Organization of Memory. New York: Academic Press, 1972.

Ward, B. and McCalla, G. Error detection and recovery in a dynamic planning environment. In the proceedings of the National Conference on Artificial Intelligence, 18-20 August 1982, in Pittsburgh, PA., 172-175.

Weinreb, D. and Moon, D. Lisp Machine Manual, Fourth Edition. Cambridge, MA: Symbolics, Inc., 1981.

Weiss, S. M. and Kulikowski, C. A. EXPERT: a system for developing consultation models. In the Proceedings of the Sixth International Joint Conference on Artificial Intelligence, 942-947, 1979.

Wickelgren, W. A. How to Solve Problems. San Francisco: W. H. Freeman and Co., 1974.

Wilensky, R. Understanding Goal-Based Stories. Research Report #140, Yale University, Department of Computer Science, New Haven, CT., 1978.

Wilensky, R. Planning and Understanding: A Computational Approach to Human Reasoning. Reading, MA: Addison-Wesley Publishing Company, 1983.

Wilensky, R., Arens, Y., and Chin, D. Talking to UNIX in English: an overview of UC. Communications of the ACM, 1984, 27, 574-593.

Wilkins, D. E. Domain-independent planning: representation and plan generation. Artificial Intelligence, 1984, 22, 269-301.

Williams, M. D. and Hollan, J. D. The process of retrieval from very long-term memory. Cognitive Science, 1981, 5, 87-119.

Winston, P. H. Learning structural descriptions from examples. In Winston (Ed.) The Psychology of Computer Vision. New York: McGraw-Hill, 1975.

Winston, P. H. Learning and reasoning by analogy. Communications of the ACM, 1980, 23, 689-703.

Winston, P. H. Artificial Intelligence second edition. Reading, MA: Addison-Wesley, 1984.

Zadeh, L. Fuzzy sets. Information and Control, 1965, 8, 338-353.

APPENDIX A

THE MEDIATOR'S CASE FILE

Organizing and representing domain knowledge is a tremendously important part of artificial intelligence research. When you succeed, it appears so obvious that it is taken for granted by those who have never made the effort. A substantial portion of the research time was spent in building a conceptual model of the domain in my head so that I would have some chance of representing the domain in the MEDIATOR program.

There are no shortages of disputes in life. So it really wasn't hard to collect many dispute examples. From the hundreds (thousands?) examined, we developed our abstract view of disputes reflected in chapter two. Most disputes are complex and involved affairs, so for our research purposes we had to simplify away most of this complexity. After all, our goal is to demonstrate the heuristic advantage of case-based reasoning as a problem solving paradigm, not to present a total computational theory of dispute mediation. The result was a set of basic canonical disputes whose variations still seemed endless. So even within these simplified disputes, we found it necessary to ignore many alternative scenarios.

Many of the 20 cases below originated in personal experience or from news reports. For example, my children, Bobby and Karen, were the source of "inspiration" for the candy dispute. Those people lucky enough to have children can identify with a parent's, usually futile, attempts to mediate domestic tranquility. Other cases were derived from examples presented in the negotiation and mediation literature. Even though I rewrote these examples, usually in order to simplify them, I am deeply indebted to the original authors and cite them where appropriate below.

CANDY DISPUTE-0

A mother is on her way home from the library when she happens on two boys standing on a street corner quarreling over a candy bar. She overhears the first little boy shout, "I want it." To which the second boy responds, "So what, I want it too." Unable to resist the opportunity to play mediator, the mother suggests that the boys divide the candy equally between them. Nodding their agreement, the boys split the candy and the mother continues homeward.

CANDY DISPUTE-1

A mother is on her way home from the library when she happens on two boys standing on a street corner quarreling over a candy bar. She overhears the first little boy shout, "I bought it, so it's mine." To which the second boy responds, "So what, if you don't give it to me I'll flatten you!" The mother stops and says to the second boy, "If he owns the candy, he does not have to give it to you." After lecturing the second boy about fighting, she continues homeward.

CANDY DISPUTE-2

A mother is on her way home from the library when she happens on two boys quarreling over a candy bar. She overhears the first little boy shout, "I want it." To which the second boy responds, "So what, I want it too." Unable to resist the opportunity to play mediator, the mother suggests that the boys divide the candy equally between them. Almost in unison, the boys reject the compromise saying, "I want the whole candy bar!" The mother thinks for a minute then suggests that they flip a coin to see who gets the candy. The boys agree and the mother continues homeward.

CANDY DISPUTE-3

A mother is on her way home from the library when she happens on two boys quarreling over a candy bar. She overhears the first little boy shout, "I want it." To which the second boy responds, "So what, I want it too." Unable to resist the opportunity to play mediator, the mother offers to help the boys settle their disagreement. The boys reluctantly agree, but with the provision that, as the first boy says, "I don't have to share it with him." With this constraint, the mother thinks for a minute then suggests that boys flip a coin to see who gets the candy. The boys agree and the mother continues homeward.

ORANGE DISPUTE-0

The mother arrives home from the library and finds her daughters quarreling over an orange. Recognizing the obvious similarity between this situation and her recent experience with the little boys, she suggests that her daughters stop quarreling and divide the orange equally between themselves by having the first daughter cut the orange into two pieces and letting the second daughter choose her piece first. The girls agree to her suggestion. The first daughter peels her half orange and eats the fruit. But her sister peels her half, throws the fruit in the trash, and uses the peel to bake a cake. (Fisher and Ury, 1981)

ORANGE DISPUTE-1

The mother went to the fruit stand to buy some more oranges. A shopper at the fruit stand was quarreling with the manager over a particular orange. The shopper said it was half the size of the others and therefore should be half the price. The manager disagreed saying that the smaller ones were more flavorful which compensated for their size. The mother suggested that they split the difference. The manager and shopper agreed and everyone seemed pleased.

BOOK DISPUTE-0

Two students came to the librarian and asked to check out the same reserved book overnight. The librarian suggested that they take turns using the book. One check it out tonight, the other tomorrow night. Which student should check it out first? This subproblem leads to a quarrel over who needs the book the worst. The librarian asks each student for their grade point average (GPA). She suggests that the student with the lowest GPA go first.

BOOK DISPUTE-1

Professors Boone and Crockett were both good friends and collectors of old books. One day they were walking to the university together, when they both spotted a few books strewn across the sidewalk in front of a small house. Boone picked up one of the volumes and was surprised to see that it was an eighteenth century printing of some Greek tragedies. Their interest aroused, the men soon discover that none of the books were printed later than 1914. About that time the door of the small house opened and a young man came out carrying another armload of books. Much to their delight, the young man gives all the books away. After calling a taxi and loading the books aboard, Boone and Crockett discuss how to divide up the books on the way back to their homes. The taxi driver overhears the professors and suggests that they each take turns choosing a book until the books are all divided. (Raiffa, 1983)

BABY DISPUTE

Two women came to Solomon both claiming to be the mother of a newborn baby. Each woman accuses the other of stealing her child as a replacement for the other's child which had been accidentally killed. There seemed to be no way to independently verify either woman's argument. Solomon said, "Divide the living child in two, and give half to the one, and half to the other." The real mother, fearing for the life of her child, begged Solomon to give the child to the second woman rather than kill it. The second woman agreed with Solomon's decision to divide the baby equally. Solomon, of course, gives the baby to the first woman. (May and Metzger, 1965, 1 Kings, 3: 16-27)

CORPSE DISPUTE

Two old women came to Solomon both claiming the remains of poor Adam, the local recluse who died last week. Much to everyone's surprise, a probate clerk had discovered that Adam was quite wealthy. In due course, his estate would become public (i.e., the King's) property unless a relative could be found. The two old women both claimed to be Adam's mother and that the other was an imposter interested only in Adam's estate. Since he could not determine who was lying, Solomon ordered that Adam's corpse be divided in half so that each woman could see to the burial of her son. As for the estate, Solomon declared that it became public property since there was no clear heir. As the old women departed, Solomon whispered instructions to one of his aides to have them followed and report back on the burial details. When Solomon later learned that the first woman had seen to all of Adam's burial because the other woman had never claimed Adam's other half, he instructed that Adam's estate be given to the first woman.

ANTARCTIC DISPUTE

Fourteen nations are meeting to decide the fate of Antarctica's natural resources. One coalition is interested in developing Antarctica's resources as a means of providing income for poorer nations. Another coalition wants Antarctica protected as a natural wilderness and object of scientific investigation.

SINAI DISPUTE

A mother reads in the paper about the Sinai dispute (before the Camp David Accords). Initially, she is reminded of the Korean War since both involve disputes over land and both involve the use of military force. Based on this reminding, she predicts that Israel and Egypt will end up dividing the Sinai equally.

She later reads that this advice was given and rejected by both Israel and Egypt. Considering that "divide equally" failed, she is reminded of her daughters' recent quarrel over an orange. She had suggested that they divide it equally, and they had rejected that, since one wanted to use the entire peel for a cake. Realizing that she hadn't taken their real goals into account, she then naturally suggested that they divide it into different parts -- one take the peel, the other the fruit. This reminding provides the suggestion that failures may occur because the goals of the disputants are misunderstood. She therefore attempts a reinterpretation of Israel and Egypt's goals.

Since Israel wants the Sinai as a military buffer zone in support of its national security, and Egypt wants the land back for its national integrity, she can now reconsider the conflict as a dispute with concordant goals. She is now reminded of the Panama Canal dispute since the disputants, the disputed object and the concordant goal relationships are similar. The analogy thus made possible guides instantiation of the "divide into different parts" plan. Using the settlement between Panama and the US, the US is replaced by Israel (the party currently in control of the object) and Panama is replaced by Egypt (the party who used to own it and wants it back). By further analogy, the prediction is made that Egypt will get economic and political control of the Sinai, while its normal right of military control will be denied.

AVOCADO DISPUTE

The mother arrives home from the library and finds her daughters quarreling over an avocado. Recognizing the obvious similarity between this situation and her recent experience with the little boys, she suggests that her daughters stop quarreling and divide the avocado equally between themselves. The second sister protests that if the mother means to literally cut the avocado in two then the seed would be ruined.

SEA DISPUTE

During the "Law of the Seas" Conference, the issue of extracting mineral and other natural resourses from the sea beds of the world effectively divided the conferees into the developed nations (those who were capable and anxious to extract these resourses) and the undeveloped nations (those who are currently unprepared to extract these resourses, but wanted to protect their future access and share of these non-renewable resources nonetheless). After much debate, the conferees agreed that the "non-territorial" waters of the world should be divided equally between the developed nations and the undeveloped nations. But this still left open the operational issue of which half of the sea beds should be assigned to which coalition. The undeveloped nations, without the technical knowledge to assess the relative value of different sea bed parcels, did not trust the developed nations to divide the sea beds fairly. Finally, it was agreed that when the developed nations are interested in a parcel of sea bed they should divide the parcel into two pieces and the undeveloped nations would choose which piece should be retained for themselves and the remaining piece assigned to the developed nations. (Raiffa, 1983)

WINDOW DISPUTE

Two men are quarreling in a library. One wants the window full open and the other wants it closed. The librarian, hearing the clamor, suggests they split the difference and open the window half way. Both men reject this suggestion, neither seemed willing to accept a compromise solution. Finally, the librarian asks the first man why he wanted the window open: "To get some fresh air." She asks the other man why he wants the window closed: "To avoid the draft." After thinking a minute she opens wide a window in the next room, bringing in fresh air without a draft. The men nod their agreement and quiet is restored to the library. (Fisher and Ury, 1981)

FARM DISPUTE

Old MacDonald has decided to sell his farm in Georgia. The Thiele Kaolin Company, which extracts kaolin from strip mines, has learned that Old MacDonald's farm has a high kaolin potential. Thiele decides to buy old MacDonald's farm. But unbeknownst to Thiele, the Georgia-Pacific Company, a lumber concern, has also decided to buy Old MacDonald's farm as a source for current and future timber. Much to Old MacDonald's delight, a bidding war develops between the two companies. After several rounds of bidding have doubled the original asking price, Thiele and Georgia-Pacific ask a realtor-mediator to help them resolve their dispute. The realtor-mediator suggests that the companies divide Old MacDonald's farm into different parts. Georgia-Pacific buys the farm but without the mining rights. Thiele buys the mining rights. First, Georgia-Pacific will harvest any current lumber from the farm's surface. Thiele then gets to mine the deforested land for its kaolin, and then restores it for use as a tree farm by Georgia-Pacific.

CONDO DISPUTE

Fred and Ethel wanted to be able to vacation in one of those fancy condominiums at the beach but couldn't afford to buy one. One evening while visiting their friends Ricky and Lucy, they mentioned their vacation dreams. Ricky suggested that the two couples form a partnership and buy a condominium to share. This seemed to be the ideal solution and both couples began working out the details. As it turned out, even in partnership with Ricky and Lucy, Fred and Ethel could only afford 25% of the purchase and maintenance costs of the condominium. So in the final arrangement, Ricky and Lucy paid 75% of all the costs. Later as the couples met with a realtor to sign the paperwork, Fred and Ethel began to draw up a schedule for the condo's use that allocated half the time to each couple. When Ricky and Lucy objected, the realtor suggested that a fair solution would be that Ricky and Lucy get to use the condo 75% of the time while Fred and Ethel use the remaining 25%. After realizing their error, Fred and Ethel apologized and began drawing up a new schedule.

HORSE DISPUTE

Joe Bob and Billy Joe were sons of Big John, one of the most famous horsemen in South Texas. Big John owned not only the most horses, but the best horses. And the best of the best was Cass Ole. Both Joe Bob and Billy Joe dearly wanted Cass Ole for his very own. One day Big John overheard the two boys fighting over who deserved Cass Ole more. Both boys claimed to be the better rider. Big John told the boys that he would settle this argument by letting the boys run a horse race, the winner would get Cass Ole. To make things fair, Big John decided that each boy could choose from a corral of horses the steed the other was to ride.

BUDGET DISPUTE

AM-JAPAN 0753
Japan Sets Limit on Military Spending
By CLYDE HABERMAN=
c. 1983 N.Y. Times News Service=

TOKYO _ The Japanese Cabinet put limits Tuesday on military spending for next year, setting in motion a fresh debate over whether the country gives its military too much or too little money.

Government officials anticipated complaints from the United States that Japan, despite planned increases, was still not providing enough funds for national defense. On the other side are domestic critics who feel that the military is getting more than its fair share at a time when the budgets of most government agencies are being slashed.

After long rounds of negotiations between the Defense Agency and the Finance Ministry, the Cabinet of Prime Minister Yasuhiro Nakasone compromised early Tuesday morning on a 6.88 percent limit on increases in military spending. It means that, at current exchange rates, the present military budget of $11.5 billion would rise to, at most, $12.3 billion in the 1984 fiscal year starting next April 1.

That 6.88 percent, however, is the ceiling; months will now be devoted to filling in specific details as to where the money should go. As is often the case, the figure ultimately approved could be smaller. In this year's budget, for example, the Cabinet originally established a limit of 7.3 percent, but that eventually was whittled down to 6.5 percent.

Among other things, Defense Agency officials worried that Tuesday's ceiling would set them back in plans for a sizable military buildup by 1988. They had been seeking an 8.9 percent increase. Originally, the Finance Ministry offered only 3.7 percent and, after negotiations, the two sides wound up splitting most of the

difference.

One issue certain to arise is whether any increase stays within 1 percent of Japan's gross national product _ a threshold established nearly 10 years ago and one that has taken on an almost mystical significance here. Despite pressures from the United States, where military spending accounts for 6.6 percent of the GNP, no recent Japanese government has been politically prepared to go over 1 percent, and officials insisted Tuesday night that any increase in 1984 would also not pierce that barrier. upi 07-12-83 06:11 ped=

VALUE DISPUTE

Adjuster: Since the dump truck was totally at fault for destroying your parked car, we have decided that the policy applies. That means you are entitled to a settlement of $3,300.
Client: I see. How did you reach that figure?
Adjuster: That's how much we decided it was worth.
Client: I understand, but what standard did you use to determine that amount? Do you know where I can buy a comparable car for that much?
Adjuster: How much are you asking for?
Client: Whatever I'm entitled to under my policy. I found a secondhand car just about like it for $3,850. Adding sales and excise tax, it comes to about $4,000.
Adjuster: $4,000! That's too much!
Client: I'm not asking for $4,000 or $3,000 or $5,000, but for fair compensation. Do you agree it's only fair I get enough to replace my car?
Adjuster: OK, I'll offer you $3,500. That's the highest I can go. It's company policy.
Client: How does the company figure that?
Adjuster: Look, $3,500 is all you'll get. Take it or leave it.
Client: $3,500 may be fair. I don't know. I certainly understand your position if you are bound by company policy. Let me ask you to find out the basis for that policy. I'll call back tomorrow at eleven, after we both have a chance to study this matter.

****

Adjuster: OK, I've got an ad here in today's paper offering a '78 Fiesta for $3,400.
Client: I see. What does it say about milage?
Adjuster: 49,000. Why?
Client: Because mine only had 25,000 miles. How many dollars does that increase the worth in your book?
Adjuster: Let's see...$150.
Client: Assuming the $3,400 as one base, that brings the figure to $3,550. Does the ad say anything about a radio?
Adjuster: No.
Client: How much extra for that in your book?
Adjuster: $125.
Client: How much for air conditioning?

****

Later that day the client picked up a check for $4,012 from the insurance adjuster. (Fisher and Ury, 1981)

APPENDIX B

MORE EXAMPLES OF THE MEDIATOR

```
(mediator candy-dispute t)
Considering the following problem:
   two boys are quarreling over a candy bar,
   which has been presented as ako M-DISPUTE.
(*DISPUTE*
 (PARTY-A (BOY1))
 (PARTY-B (BOY2))
 (DISPUTED-OBJ CANDY1)
 (ARGUMENT-A (*ARGUMENT* (ARGUER (BOY1))
                         (SUPPORT (*PHYS-CONTROL* (ACTOR BOY1)
                                                  (OBJECT CANDY1)))))
 (ARGUMENT-B (*ARGUMENT* (ARGUER (BOY2))
                         (SUPPORT (*PHYS-CONTROL* (ACTOR CHILD)
                                                  (OBJECT CANDY1))))))
ATTEMPTING TO RECALL SIMILAR PROBLEMS IN ORDER TO CLASSIFY THIS ONE...
  looking for disputes with similar disputants or with similar goals...
  looking for disputes whose disputants used similar arguments.
  looking for disputes involving similar objects...
There were zero previous cases found.
Given that there are no similar cases,
  will use a default context classification.

This dispute will be referred to as #<M-PHYS-DISPUTE 5347161>
(*PHYS-DISPUTE*
 (PARTY-A (BOY1))
 (PARTY-B (BOY2))
 (DISPUTED-OBJ CANDY1)
 (ARGUMENT-A (*ARGUMENT* (ARGUER (BOY1))
                         (SUPPORT (*PHYS-CONTROL* (ACTOR BOY1)
                                                  (OBJECT CANDY1)))))
 (ARGUMENT-B (*ARGUMENT* (ARGUER (BOY2))
                         (SUPPORT (*PHYS-CONTROL* (ACTOR CHILD)
                                                  (OBJECT CANDY1))))))
BOY1 has presented an argument recognized as type *PHYS-CONTROL*
  which is normally presented in an attempt to pursuade by force.
Therefore no inferences will be based on BOY1's argument.

BOY2 has presented an argument recognized as type *PHYS-CONTROL*
  which is normally presented in an attempt to pursuade by force.
Therefore no inferences will be based on BOY2's argument.

Using elaboration to infer BOY1's goal
   from normal uses of the disputed object in this context...
    normal use of -#<M-CANDY 5341331>- is being assumed.
Therefore BOY1 is inferred to have a M-INGEST goal.
Using elaboration to infer BOY2's goal
   from normal uses of the disputed object in this context...
    normal use of -#<M-CANDY 5341331>- is being assumed.
Therefore BOY2 is inferred to have a M-INGEST goal.
(*PHYS-DISPUTE*
 (PARTY-A (BOY1 (*GOAL* (*INGEST* (ACTOR BOY1) (OBJECT CANDY1)))))
 (PARTY-B (BOY2 (*GOAL* (*INGEST* (ACTOR BOY2) (OBJECT CANDY1)))))
 (DISPUTED-OBJ CANDY1)
 (ARGUMENT-A
  (*ARGUMENT* (ARGUER (BOY1 (*GOAL* (*INGEST* (ACTOR BOY1)
                                              (OBJECT CANDY1)))))
              (SUPPORT (*PHYS-CONTROL* (ACTOR BOY1)
                                       (OBJECT CANDY1)))))
 (ARGUMENT-B
  (*ARGUMENT* (ARGUER (BOY2 (*GOAL* (*INGEST* (ACTOR BOY2)
                                              (OBJECT CANDY1)))))
              (SUPPORT (*PHYS-CONTROL* (ACTOR CHILD)
                                       (OBJECT CANDY1))))))
Goal relationship is inferred to be COMPETITION.

ATTEMPTING TO SELECT A NEGOTIATION PLAN
   TO RESOLVE THE DISPUTE IDENTIFIED AS #<M-PHYS-DISPUTE 5347161>.
  looking for disputes with similar disputants or with similar goals...
  looking for disputes whose disputants used similar arguments.
  looking for disputes involving similar objects...
There were zero previous cases found.
```

```
Using default reasoning to select a resolution plan.
Considering plans normally useful for M-PHYS-DISPUTE disputes.
Looking at the plan called #<M-DIVIDE-EQUALLY 5347450>
   which appears to be applicable.
My reasoning is as follows:
   This plan has not failed to my knowledge on similar previous problems
   It normally doesn't make sense to share CANDY1,
   since its functionality is destroyed by its consumption,
   CANDY1 can be divided without loss of functionality;
   when this is considered with a compromise planning policy
   and my inference that the parties' goals are in competition;
   all indicate that divide equally is a reasonable plan.
TRYING TO SPECIALIZE DIVIDE EQUALLY BY DEFAULT REASONING.
Looking at the plan called split the difference
   which does not seem applicable.
Considering the plan called one cuts the other chooses
   which appears to be applicable.
My additional reasoning is as follows:
   This plan has not failed to my knowledge on similar previous problems
   The fact that CANDY1 can be split without destruction
   and my classification of this dispute as a M-PHYS-DISPUTE
   indicate to me that one cuts the other chooses is a reasonable plan.

I suggest that the plan called one cuts the other chooses be used.
INSTANTIATING EXPECTED CONTRACT RESULTING FROM PLAN APPLICATION.
  Using default reasoning since I have no previous
     experience with this plan.
PREDICTIONS ASSOCIATED WITH USING ONE CUTS THE OTHER CHOOSES FOR THE
  CURRENT DISPUTE KNOWN AS #<M-PHYS-DISPUTE 5347161> ARE EMBODIED IN THE
  CONTRACT KNOWN AS #<M-DIVIDED-OBJ-CONTRACT 5350273>
Do you agree, that this is the best solution? (Y or N) Yes.
Do you know the results of the plan's execution? (Y or N) Yes.

Please indicate the results: results-ok

You said:
((*INGEST* (ACTOR BOY1) (OBJECT (HALF CANDY1)))
 (*INGEST* (ACTOR BOY2) (OBJECT (HALF CANDY1))))
Which matches my expectations, therefore
   this is a successful problem solving experience.
#<M-MEMORY 5351001>

(mediator orange-dispute t)
Considering the following problem:
   two sisters are quarreling over an orange,
   which has been presented as ako M-DISPUTE.
(*DISPUTE*
 (PARTY-A (SISTER1))
 (PARTY-B (SISTER2))
 (DISPUTED-OBJ ORANGE1)
 (ARGUMENT-A (*ARGUMENT* (ARGUER (SISTER1))
                         (SUPPORT (*PHYS-CONTROL* (ACTOR SISTER1)
                                                  (OBJECT ORANGE1)))))
 (ARGUMENT-B
  (*ARGUMENT* (ARGUER (SISTER2))
              (SUPPORT (*PHYS-CONTROL* (ACTOR SISTER2)
                                       (OBJECT ORANGE1))))))
ATTEMPTING TO RECALL SIMILAR PROBLEMS IN ORDER TO CLASSIFY THIS ONE...
  looking for disputes with similar disputants or with similar goals...
  looking for disputes whose disputants used similar arguments.
  looking for disputes involving similar objects...
   reminded of two boys are quarreling over a candy bar
    because the object in that case, CANDY1,
    is the only other object in my experience.
   reminded of two boys are quarreling over a candy bar
      because the argument used in that case, the possession argument,
                        was of the same type, M-POSSESS, as this dispute.
There was one previous case found.
   #<M-PHYS-DISPUTE 22131722> was the case where two boys
     are quarreling over a candy bar.
Selected the dispute where two boys are quarreling over a candy bar
   which was classified as M-PHYS-DISPUTE for further consideration.
Transferring previous classification to this dispute.

This dispute will be referred to as #<M-PHYS-DISPUTE 22475211>
(*PHYS-DISPUTE*
 (PARTY-A (SISTER1))
 (PARTY-B (SISTER2))
 (DISPUTED-OBJ ORANGE1)
```

```
 (ARGUMENT-A (*ARGUMENT* (ARGUER (SISTER1))
                         (SUPPORT (*PHYS-CONTROL* (ACTOR SISTER1)
                                                  (OBJECT ORANGE1)))))
 (ARGUMENT-B
  (*ARGUMENT* (ARGUER (SISTER2))
              (SUPPORT (*PHYS-CONTROL* (ACTOR SISTER2)
                                       (OBJECT ORANGE1))))))
SISTER1 has presented an argument recognized as type *PHYS-CONTROL*
  which is normally presented in an attempt to pursuade by force.
Therefore no inferences will be based on SISTER1's argument.

SISTER2 has presented an argument recognized as type *PHYS-CONTROL*
  which is normally presented in an attempt to pursuade by force.
Therefore no inferences will be based on SISTER2's argument.

Attempting to transfer goal type from recalled case
   #<M-PHYS-DISPUTE 22131722>
   checking for consistency with normal uses of ORANGE1.
Thus SISTER1 is inferred to have a M-INGEST goal
   which is consistent with the normal uses of ORANGE1 in this context.

Attempting to transfer goal type from recalled case
   #<M-PHYS-DISPUTE 22131722>
   checking for consistency with normal uses of ORANGE1.
Thus SISTER2 is inferred to have M-INGEST
   which is consistent with the normal uses of #<M-ORANGE 22123746>
   in this context.
(*PHYS-DISPUTE*
 (PARTY-A (SISTER1 (*GOAL* (*INGEST* (ACTOR SISTER1)
                                     (OBJECT ORANGE1)))))
 (PARTY-B (SISTER2 (*GOAL* (*INGEST* (ACTOR SISTER2)
                                     (OBJECT ORANGE1)))))
 (DISPUTED-OBJ ORANGE1)
 (ARGUMENT-A
  (*ARGUMENT* (ARGUER (SISTER1 (*GOAL* (*INGEST* (ACTOR SISTER1)
                                                 (OBJECT ORANGE1)))))
              (SUPPORT (*PHYS-CONTROL* (ACTOR SISTER1)
                                       (OBJECT ORANGE1)))))
 (ARGUMENT-B
  (*ARGUMENT*
   (ARGUER (SISTER2 (*GOAL* (*INGEST* (ACTOR SISTER2)
                                      (OBJECT ORANGE1)))))
   (SUPPORT (*PHYS-CONTROL* (ACTOR SISTER2) (OBJECT ORANGE1))))))
Goal relationship is inferred to be COMPETITION.

ATTEMPTING TO SELECT A NEGOTIATION PLAN
   TO RESOLVE THE DISPUTE IDENTIFIED AS #<M-PHYS-DISPUTE 22475211>.

Using previously recalled case, where two boys are quarreling
   over a candy bar.
   It was resolved using the plan known as one cuts the other chooses.
Checking for applicability of that plan to the current case..

My additional reasoning is as follows:
   This plan has not failed to my knowledge on similar previous problems
   The fact that ORANGE1 can be split without destruction
   and my classification of this dispute as a M-PHYS-DISPUTE
   indicate to me that one cuts the other chooses is a reasonable plan.
Selecting the plan one cuts the other chooses for this dispute
   and instantiating.

I suggest that the plan called one cuts the other chooses be used.
INSTANTIATING EXPECTED CONTRACT RESULTING FROM PLAN APPLICATION.

ATTEMPTING TO RECALL PREVIOUS EXPERIENCE WITH ONE CUTS THE OTHER CHOOSES
  reconstructing my previous experience with this plan results in a
     contract identified as #<M-DIVIDED-OBJ-CONTRACT 22133144>
  checking the applicability of this contract to this situation ...

Evaluating old contract based on the four invariance
   features disputant goals, dispute type, disputed object, and disputants.

With a rating of 15 out of 17,
   the old contract is judged acceptable based on these criteria.

  using it to guide current contract construction ...
   matching SISTER1 with BOY1 ...
   matching SISTER2 with BOY2...
   matching ORANGE1 with CANDY1...
```

```
   matching
(*GOAL* (*INGEST* (ACTOR SISTER1) (OBJECT ORANGE1)))T with
(*HALF* CANDY1)T...
   matching
(*GOAL* (*INGEST* (ACTOR SISTER2) (OBJECT ORANGE1)))T with
(*HALF* CANDY1)T...
   transferring other components of contract unchanged.
PREDICTIONS ASSOCIATED WITH USING ONE CUTS THE OTHER CHOOSES FOR THE
  CURRENT DISPUTE KNOWN AS #<M-PHYS-DISPUTE 22475211> ARE EMBODIED IN THE
  CONTRACT KNOWN AS #<M-DIVIDED-OBJ-CONTRACT 22476106>
Do you agree, that this is the best solution? (Y or N) Yes.
Do you know the results of the plan's execution? (Y or N) Yes.

Please indicate the results: internal-intentional

You said:
((*INGEST* (ACTOR SISTER1)
           (OBJECT (HALF ORANGE1))
           (INST (*PREPARE* (ACTOR SISTER1) (OBJECT (HALF ORANGE1)))))
 (*PREPARE* (ACTOR SISTER2)
            (OBJECT CAKE1)
            (INST (*PHYS-CONTROL* (ACTOR SISTER2) (OBJECT PEEL1)))))
This does not match my expectations, which were:
((*INGEST* (ACTOR SISTER1) (OBJECT (HALF ORANGE1)))
 (*INGEST* (ACTOR SISTER2) (OBJECT (HALF ORANGE1))))
Even though this resolution was accepted,
  I will attempt to explain this expectation failure
  and see if a better resolution can be found.
Considering the following problem:
   failed negotiation for two sisters are quarreling over an orange,
   which has been presented as ako M-UNSUCCESSFUL-NEGOTIATION.
#<M-UNSUCCESSFUL-NEGOTIATION 22476701>
ATTEMPTING TO RECALL SIMILAR PROBLEMS IN ORDER TO CLASSIFY THIS ONE...

ATTEMPTING TO RECALL SIMILAR FAILURES ...
  looking for previous negotiation plan failures...

I do not recall any previous failures.

Given that there are no similar failed cases,
  will use default reasoning for failure context classification.

INFERRING A FAILURE OF THE GOAL INFERENCE TYPE BECAUSE
  SISTER2'S ACTION INDICATES AN ALTERNATE GOAL.
This failure will be referred to as #<M-WRONG-GOAL-INFERENCE 22477222>

SISTER1 is represented as having the goal #<M-INGEST 22475431>

(*GOAL* (*INGEST* (ACTOR SISTER1) (OBJECT ORANGE1)))T

SISTER2 is represented as having the goal #<M-INGEST 22475445>

(*GOAL* (*INGEST* (ACTOR SISTER2) (OBJECT ORANGE1)))T

ATTEMPTING TO SELECT A REMEDY FOR
   THE FAILURE IDENTIFIED AS #<M-WRONG-GOAL-INFERENCE 22477222>.

ATTEMPTING TO RECALL SIMILAR FAILURES ...
  looking for previous negotiation plan failures...

I do not recall any previous failures.

Using default reasoning to select a remedy.
Considering remedies normally useful for M-WRONG-GOAL-INFERENCE failures.
Looking at the remedy called #<M-USE-ACTUAL-EVENTS 22477476>
   which appears to be applicable.
Using feedback to consider other possible goal interpretations
   for the current case ...
I previously inferred that SISTER2 had a goal of type M-INGEST.
Based on the results, I should have inferred a *PREPARE* type goal.
In addition, a different part,  was used.
Therefore SISTER2's goal representation will be changed to reflect this.

Remediation complete.

Given this new information, I'll reconsider this problem.
Considering the following problem:
   two sisters are quarreling over an orange,
   which has been presented as ako M-PHYS-DISPUTE.
```

```
(*PHYS-DISPUTE*
 (PARTY-A (SISTER1 (*GOAL* (*INGEST* (ACTOR SISTER1)
                                     (OBJECT ORANGE1)))))
 (PARTY-B
  (SISTER2
   (*GOAL*
    (*PREPARE* (ACTOR SISTER2)
               (OBJECT CAKE1)
               (INST (*GOAL* (*PHYS-CONTROL* (ACTOR SISTER2)
                                             (OBJECT PEEL1))))))))
 (DISPUTED-OBJ ORANGE1)
 (ARGUMENT-A
  (*ARGUMENT* (ARGUER (SISTER1 (*GOAL* (*INGEST* (ACTOR SISTER1)
                                                 (OBJECT ORANGE1)))))
              (SUPPORT (*PHYS-CONTROL* (ACTOR SISTER1)
                                       (OBJECT ORANGE1)))))
 (ARGUMENT-B
  (*ARGUMENT*
   (ARGUER
    (SISTER2
     (*GOAL*
      (*PREPARE* (ACTOR SISTER2)
                 (OBJECT CAKE1)
                 (INST (*GOAL* (*PHYS-CONTROL* (ACTOR SISTER2)
                                               (OBJECT PEEL1)))))))))
   (SUPPORT (*PHYS-CONTROL* (ACTOR SISTER2) (OBJECT ORANGE1)))))
This dispute will be referred to as #<M-PHYS-DISPUTE 22475211>
(*PHYS-DISPUTE*
 (PARTY-A (SISTER1 (*GOAL* (*INGEST* (ACTOR SISTER1)
                                     (OBJECT ORANGE1)))))
 (PARTY-B
  (SISTER2
   (*GOAL*
    (*PREPARE* (ACTOR SISTER2)
               (OBJECT CAKE1)
               (INST (*GOAL* (*PHYS-CONTROL* (ACTOR SISTER2)
                                             (OBJECT PEEL1))))))))
 (DISPUTED-OBJ ORANGE1)
 (ARGUMENT-A
  (*ARGUMENT* (ARGUER (SISTER1 (*GOAL* (*INGEST* (ACTOR SISTER1)
                                                 (OBJECT ORANGE1)))))
              (SUPPORT (*PHYS-CONTROL* (ACTOR SISTER1)
                                       (OBJECT ORANGE1)))))
 (ARGUMENT-B
  (*ARGUMENT*
   (ARGUER
    (SISTER2
     (*GOAL*
      (*PREPARE* (ACTOR SISTER2)
                 (OBJECT CAKE1)
                 (INST (*GOAL* (*PHYS-CONTROL* (ACTOR SISTER2)
                                               (OBJECT PEEL1)))))))))
   (SUPPORT (*PHYS-CONTROL* (ACTOR SISTER2) (OBJECT ORANGE1)))))
SISTER1 has presented an argument recognized as type *PHYS-CONTROL*
  which is normally presented in an attempt to pursuade by force.
Therefore no inferences will be based on SISTER1's argument.

SISTER2 has presented an argument recognized as type *PHYS-CONTROL*
  which is normally presented in an attempt to pursuade by force.
Therefore no inferences will be based on SISTER2's argument.

SISTER1 is represented as having the goal #<M-INGEST 22475431>

(*GOAL* (*INGEST* (ACTOR SISTER1) (OBJECT ORANGE1)))T

SISTER2 is represented as having the goal #<M-PREPARE 22477535>

(*GOAL* (*PREPARE* (ACTOR SISTER2)
                   (OBJECT CAKE1)
                   (INST (*GOAL* (*PHYS-CONTROL* (ACTOR SISTER2)
                                                 (OBJECT PEEL1))))))T
(*PHYS-DISPUTE*
 (PARTY-A (SISTER1 (*GOAL* (*INGEST* (ACTOR SISTER1)
                                     (OBJECT ORANGE1)))))
 (PARTY-B
  (SISTER2
   (*GOAL*
    (*PREPARE* (ACTOR SISTER2)
               (OBJECT CAKE1)
```

```
              (INST (*GOAL* (*PHYS-CONTROL* (ACTOR SISTER2)
                                            (OBJECT PEEL1)))))))
 (DISPUTED-OBJ ORANGE1)
 (ARGUMENT-A
  (*ARGUMENT* (ARGUER (SISTER1 (*GOAL* (*INGEST* (ACTOR SISTER1)
                                                 (OBJECT ORANGE1)))))
              (SUPPORT (*PHYS-CONTROL* (ACTOR SISTER1)
                                       (OBJECT ORANGE1)))))
 (ARGUMENT-B
  (*ARGUMENT*
   (ARGUER
    (SISTER2
     (*GOAL*
      (*PREPARE* (ACTOR SISTER2)
                 (OBJECT CAKE1)
                 (INST (*GOAL* (*PHYS-CONTROL* (ACTOR SISTER2)
                                               (OBJECT PEEL1)))))))))
   (SUPPORT (*PHYS-CONTRDL* (ACTOR SISTER2) (OBJECT ORANGE1))))))
Goal relationship is inferred to be CONCORDANT.

ATTEMPTING TO SELECT A NEGOTIATION PLAN
   TO RESOLVE THE DISPUTE IDENTIFIED AS #<M-PHYS-DISPUTE 22475211>.
  looking for disputes with similar disputants or with similar goals...
  looking for disputes whose disputants used similar arguments.
  looking for disputes involving similar objects...
   reminded of two boys are quarreling over a candy bar
      because in that case BOY1 also had a goal of type M-INGEST.
   reminded of two boys are quarreling over a candy bar
    because the object in that case, CANDY1, is the only other object
    in my experience.
   reminded of two boys are quarreling over a candy bar
      because the argument used in that case, the possession argument,
                          was of the same type, M-POSSESS, as this dispute.
There was one previous case found.
   #<M-PHYS-DISPUTE 22131722> was the case where two boys are quarreling
     over a candy bar.
Using default reasoning to select a resolution plan.
Considering plans normally useful for M-PHYS-DISPUTE disputes.

Looking at the plan called #<M-DIVIDE-EQUALLY 22500005>
   which does not seem applicable.
Looking at the plan called #<M-TAKE-TURNS 22500337>
   which does not seem applicable.
Looking at the plan called #<M-DIVIDE-UNEQUALLY 22500525>
   which appears to be applicable.
My reasoning is as follows:
   none of the previous plans are applicable and
   one of the preconditions for unequal division
   is satisfied which indicate to me that unequal division is possible.
TRYING TO SPECIALIZE DIVIDE UNEQUALLY BY DEFAULT REASONING.
Considering the plan called divide into different parts which appears
   to be applicable.
My reasoning is as follows:
   This plan has not failed to my knowledge on similar previous problems
   ORANGE1 can be divided without destruction
   when this is considered with my initial classification of this
   dispute as a M-PHYS-DISPUTE
   and my inference that the parties' goals are concordant all indicate
   to me that division into different parts will satisfy everyone.

I suggest that the plan called divide into different parts be used.
INSTANTIATING EXPECTED CONTRACT RESULTING FROM PLAN APPLICATION.
  Using default reasoning since I have no previous experience with this plan.
PREDICTIONS ASSOCIATED WITH USING DIVIDE INTO DIFFERENT PARTS FOR THE
  CURRENT DISPUTE KNOWN AS #<M-PHYS-DISPUTE 22475211> ARE EMBODIED IN THE
  CONTRACT KNOWN AS #<M-DIFF-PARTS-CONTRACT 22501242>
Do you agree, that this is the best solution? (Y or N) Yes.
Do you know the results of the plan's execution? (Y or N) Yes.

Please indicate the results: results-ok

You said:
((*INGEST* (ACTOR SISTER1)
           (OBJECT (HALF ORANGE1))
           (INST (*PREPARE* (ACTOR SISTER1) (OBJECT (HALF ORANGE1)))))
 (*PREPARE* (ACTOR SISTER2)
            (OBJECT CAKE1)
            (INST (*PHYS-CONTROL* (ACTOR SISTER2) (OBJECT PEEL1)))))
Which matches my expectations,
```

therefore this is a successful problem solving experience.

Specializing memory node of type M-SUCCESSFUL-NEGOTIATION
with knowledge of type M-SUCCESSFUL-RENEGOTIATION

Generalizing M-FOOD from the integration of M-ORANGE with M-CANDY

Generalizing M-PERSON from the integration of M-GIRL with M-BOY

Generalizing M-PHYSICAL-CONTROL from the integration of M-PREPARE
with M-INGEST

Generalizing M-FOOD from the integration of M-CAKE with M-CANDY

Generalizing M-PERSON from the integration of M-GIRL with M-BOY

Generalizing M-FOOD from the integration of M-ORANGE with M-CANDY

Generalizing M-PERSON from the integration of M-GIRL with M-BOY

Generalizing M-PHYSICAL-CONTROL from the integration of M-PREPARE
with M-INGEST

Generalizing M-FOOD from the integration of M-CAKE with M-CANDY

Generalizing M-PERSON from the integration of M-GIRL with M-BOY

Generalizing M-FOOD from the integration of M-ORANGE with M-CANDY

Generalizing M-FOOD from the integration of M-ORANGE with M-CANDY

Generalizing M-PERSON from the integration of M-GIRL with M-BOY

Updating memory for my previous failed effort at resolution.

Generalizing M-RESOLUTION from the integration of M-REMEDIATION
with M-SUCCESSFUL-NEGOTIATION

Finding Creative Solutions in Adversarial Impasses*

Katia Sycara

School of Information and Computer Science
Georgia Institute of Technology
Atlanta, Georgia 30332
Telephone: (404) 894-5550
Computer address: katia@ gatech (CSNET)

ABSTRACT

This paper presents a method for generating creative solutions to resolve adversarial impasses that makes use of memory structures based on goal interactions and blame attribution. These knowledge structures are called Situational Assessment Packets (SAPs). SAPs contain general strategies for satisfying multiple conflicting goals either totally or partially. These resolution strategies are evaluated for applicability to a situation by considering interactions of the goals of the problem solver and the goals of the interacting agents. SAPs also provide advice about preventing and recovering from failures and justifications for a proposed solution. This work is part of the PERSUADER, a computer program that functions as a third party problem solver (mediator) in hypothetical labor negotiations.

February 23, 1987

Topic area: Cognitive Modeling
Science track
Key Words: Adversarial Reasoning, Multi-agent Planning, Partial Goal Satisfaction

* This work was supported by ARO Grant No. DAAG 29-85-K-0023.

Finding Creative Solutions in Adversarial Impasses

Katia Sycara

School of Information and Computer Science
Georgia Institute of Technology
Atlanta, Georgia 30332
Telephone: (404) 894-5550
Computer address: katia@ gatech (CSNET)

1. Introduction

> Event 1: The Blackhound company is in contract negotiations with its union. The union's demands are a 20 percent wage increase, a 6 percent increase in pensions, and seniority as the sole determinant of promotions and layoffs. The Blackhound company's initial position is a 3 percent wage increase, a 1 percent increase in pensions, and the criteria for promotions and layoffs to be determined solely by the company. The parties refuse to move from their positions and as a strike deadline approaches a mediator is called in.
>
> After an incremental shifting of positions, the parties agree to the following proposal of the mediator: 8 percent wage increase, 4 percent increase in pensions, promotions and layoffs to be governed by seniority and ability with ability receiving a higher weight.
>
> Event 2: During contract negotiations, Southern Airlines presents its employees with the ultimatum that if they don't take wage cuts of 8%, the company which has become non-competitive, will have to go bankrupt. The employees protest demanding wage increases and a mediator is called in. The mediator finds out that Southern Airlines has been loosing money because of mismanagement in an industry where other airlines are making money. She proposes that the employees accept 6% wage cuts and that the company give stock to the employees as well as have employee representatives sit on the board of directors.

Events 1 and 2 deal with adversarial situations. In adversarial situations the goals of two or more disputants are in conflict. In the above Events, goals of a union and a company are in conflict. Both Events illustrate a resolution by compromise of these conflicting goals. There is a main difference, however. The solution in Event 1 is a perturbation of the values of the adversaries' original goals. The solution in Event 2, on the other hand, is creative. It is not merely the result of modifications of the values of the input goals, but new elements not predicted by the input also enter the solution: stock options for the employees, and employee representation on the board of directors.

While an event such as Event 1 is fairly typical and therefore requires fairly shallow reasoning, Event 2 is far from typical (companies in a prosperous industry do not usually lose money) and requires creative problem solving. In cases where the situation is novel, a problem solver has to take into account, not only the apparent goals of the agents, but also higher level goals and their interactions. In general, cases that require creative problem solving are those that violate expectations about (a) prevailing practice, namely what the situation of similar agents is (b) role themes of the interacting agents, (c) beliefs about the rationality of the agents,* (d) beliefs about the temporal continuation of a state. Event 2 is an example of expectation violation arising from prevailing practice.

We propose a high level knowledge structure, called a Situational Assessment Packet (SAP) that captures the abstract structure of an atypical problem solving situation involving interacting agents in terms of the expectation violations that are involved. SAPs are like MOPs (Kolodner, 1984; Schank, 1982) in that they both store generalized information and are memory organizing structures. They are also like TOPs (Schank, 1982) in that they organize expectations and explanations of expectation failures in a domain-independent manner. Once a SAP has been accessed, the problem solver is provided with a source of advice about the causal relationships that are active in the problem at hand. In addition, SAPs warn the problem solver about potential failures, so that these failures can be avoided. SAPs also contain planning strategies and justifications that can be used in constructing an acceptable solution. The SAP that captures the situation in Event 2 is MISMANAGEMENT of some resource by one of the agents resulting in endangering a common high level goal of the agents'. The SAP also contains advice about generating an acceptable solution based on equity, namely have the guilty party bear the brunt of the cost. The SAP also contains information about construcing a justification for the proposed solution.

* By rationality we mean an agent's reluctance to follow a course of action that will result in loss of benefits.

1.1. Situational Assessment Packets (SAPs)

In adversarial situations a problem solver has to come up with a solution that satisfies multiple goals simultaneously. Such a solution has to take into consideration the competitive nature of the agents' goals and interactions between goals and plans. In such situations the description of a problem in terms of goals and plans alone forces the problem solver in considering specific and separate plans for the satisfaction of each goal. This is not only wasteful but also deprives the problem solver of the opportunity to access "analogous" plans that exhibit similarity between the *interactions* of the goals (plans) rather than similarity between the goals (plans) themselves. For example, such a planner would not be able to recognize the similarity of Event 2 to the following event:

> Event 3: Tom and Jerry are project partners in a computer science class. Tom finishes his part of the project in time, but Jerry starts his part the night before the project is due and does a lousy job. The teacher assigns separate grades to the two parts of the project, though this is not his customary practice.

Event 3 is an example of the SAP MISMANAGEMENT since Jerry mismanages time with the result of jeopardizing the success of the joint project. Experience with planning strategies in event 3 would be useful for a planner dealing with event 2.

SAPs also serve as episodic memory structures since they organize cases that involve similar expectation violations. SAP MISMANAGEMENT, for example, organizes cases where, due to mismanagement of some resource by one partner, the partners' common goal is threatened with failure. Such cases include Events 2 and 3, cases where an officer of an organization (e.g., a union, a church) has mismanaged funds, and cases where a military leader has mismanaged his part of a campaign.

The ability to store cross-contextual episodes/cases makes SAPs very powerful mechanisms. Once problem solving advice has been learned in one context, it can help processing in a different context, if the experience was recognized in terms of an appropriate SAP.

In general, SAPs are accessed when the case under consideration deviates from a typical one. They contain an abstracted structure representing situation-outcome patterns in terms of: (1) a problem solving situation, (2) expectations associated with the situation, (3) the reason

the expectation is violated (4) who/what is responsible for the violation, (5) how a third party problem solver can find an equitable solution, and (6) how to justify the solution. The fact that an equitable solution is sought acts as a constraint on the possible solutions available to the problem solver.

1.2. SAP examples

If we abstract the problem solving structure out of Events 2 and 3, we get the following SAP:

SAP MISMANAGEMENT

recognition criteria:

(a) x and y have a non-competitive high level goal G

(b) x mismanages some resource that is an enablement condition C for the achievement of G.

(c) G is in danger of failing

solution:

an equitable solution to prevent the failure of G is to have x, the guilty party, bear the brunt of the recovery cost

justification:

appeal to theme of fairness and add that if y does not perceive the solution as just, then y will not cooperate and thus G will fail (which certainly x does not want).

In Event 2, the non-competitive goal that the company and union share is to prevent the company from going bankrupt, since that would hurt both the company and the union. In Event 3, the non-competitive goal of Tom and Jerry is to get a good grade on their project. One characteristic of SAP MISMANAGEMENT is that one of the interacting human agents is to blame for the danger of the failure of the common goal. This failure is used as the justification for the proposed solution, which would certainly be disagreeable to the guilty party. A problem solver using this SAP appeals to the theme of "just desserts", which is used both as a justification and a persuasive argument for the acceptance of the solution. The subject of persuasive argumentation is treated in more detail in (Sycara, 1985).

In other problem solving situations, an actor finds himself facing a dilemma. Loyalty to another actor with whom the first is bound through a thematic relationship, causes the dilemma. A third party problem solver is asked to give advice regarding this dilemma. Consider the following events:

Event 4: A local union gives high priority to pensions although the majority of its members are young with the result of making agreement during contract negotiations impossible. The mediator that handles the case finds out that this union attitude is due to its desire to follow the guidelines of the international union whose program mandates as a negotiation goal high pension increases.

Event 5: Susan is torn between her love for John and wish to marry him and her family's opposition because of disapproval of his lifestyle and political views. A friend tells her that if her family really cared for her they would respect her wishes.

The above two situations are captured by SAP MISPLACED-LOYALTY.

SAP MISPLACED-LOYALTY

recognition criteria:

(a) x goes against his interest in order to conform to third party's wishes

(b) this endangers an achievement goal G of x

(c) third party is blamed

solution:

suggestion to look after own interest

justification:

point out contradictory attitude of third party towards x

Other SAPs we have identified include UNFORESEEN-DISASTER (external chance events force the agents into unpleasant situations), LAME-DUCK (an agent has the title of an office but not the authority), and DETRIMENTAL-PREROGATIVES (a prerogative is no longer advantageuos to the grantor because of changing circumstances). A full presentation of SAPs can be found in (Sycara, 1987).

1.3. SAP Recognition

To use SAPs in problem solving, they must be recognized or triggered at appropriate times. Because they describe abstract situations, SAPs are triggered as a result of abstract goal/blame/plan analysis. This analysis is based on (1) expectations that the problem solving situation violates, and (2) the cause of the violation (who is to blame).

When a reasoner is faced with a problem solving situation, he usually has some expectations that pertain in the situation. The violation of these expectations indicates the potential existence of a SAP. SAP recognition rules are associated with each expectation violation category (presented in the introduction). These rules guide the reasoner's search for blame

attribution. Once blame attribution has been performed, the appropriate SAP is uniquely identified.

If SAPs were recognized only after actual failures, their use in problem solving would be limited. They would not be able to warn a problem solver about potential failures. A major component of SAP recognition involves analyzing the given situation for potential errors at each point where the problem solver has to make a suggestion. Once the features of a situation have alerted a reasoner to the possible existence of a SAP, more processing is done for recognizing which is the appropriate SAP. If such a SAP is recognized, then the warning advice present is at the disposal of the reasoner. This advice is *preventive* rather than advice for *recovery* once a problem has already arisen. Consider, for example the following situations:

> Event 6: The WINO bottling company is very sensitive about management rights. Any time the union proposes contract language that the company perceives as a threat to management rights, the company grants the union big economic concessions to avoid the language change. WINO becomes noncompetitive as a result of increased labor costs and goes bankrupt.
>
> Event 7: A chemical company is in contract negotiations. The union has proposed changes in work rules that the company refuses. The company suggests to the mediator that it is willing to grant higher economic benefits, if the union will withdraw its demand for work changes. The mediator warns the company that if it makes this into a practice, it risks becoming non-competitive.

Event 7 is an illustration of a SAP being used as a source of preventive advice. The mediator recognizes the possible existence of SAP IDEOLOGY. This SAP characterizes a conflict situation where an agent places an unrealistically high value on a goal because of ideological motivations. SAP IDEOLOGY is recognized in Event 7 because the behavior of the company violates the mediator's expectations about economic rationality. Indexed under this SAP is Event 6 which the mediator accesses.

The SAP recognition process behaves as follows: If the entity to be blamed is present in the input, the SAP into which the current episode fits is uniquely recognized. Inferences and suggestions associated with this SAP are performed. If, on the other hand, the input indicates only an expectation violation, then the presence of a SAP is suspected and the cause of the violation is sought. This leads to an attempt at blame attribution.

1.4. Causal Structure of SAPs

The internal causal structure of a SAP guides the problem solver in proposing a solution. The causal structure of a SAP is a graph whose nodes represent goals, states and actions of the agents. These entities are connected via various types of links. A link between two goals, for example, denotes whether one goal is instrumental to the other; a link between an action and a goal (or a state) denotes whether the action is a precondition of the goal (or results in the state). As an illustration, let us consider the SAP MISPLACED-LOYALTY. The abstract causal structure of this SAP is as follows:

Causal structure of SAP MISPLACED-LOYALTY

Actor x has an achievement goal G1 (the primary goal).
G1 is an instrumental goal to some higher level thematic goal G0.
x also has another (secondary) achievement goal G2, ACHIEVE-LOYALTY
to actor y (G2 characterizes this SAP).
Precondition P1 for the achievement of G2 is OBEY(x, y).
Actor y has a BELIEF-STATE S1 which has as consequence BELIEF-STATE S2.
S2 causes y to want x to abandon G1.

The following figure shows the abstract causal structure of SAP MISPLACED-LOYALTY.

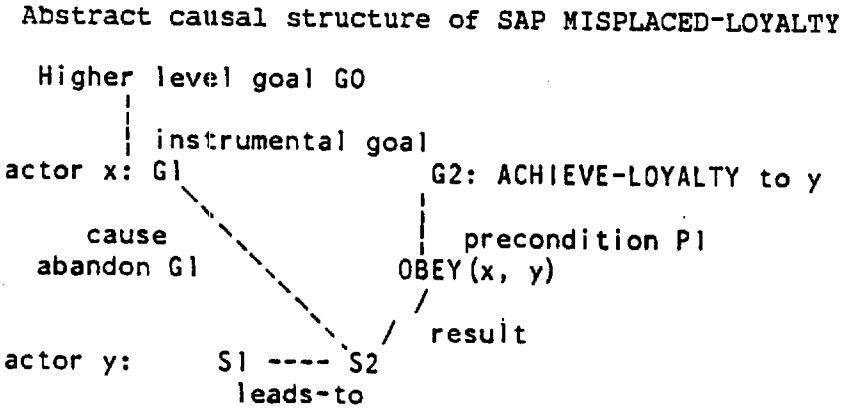


Figure 1

The nodes in the figure represent goals, states and actions of the actors x and y. The link labelled "instrumental goal" connects G1 to the higher level goal G0 of actor x. The link between states S1 and S2 denotes that S1 is the cause (or leads-to) S2. The link between S2 and G1 signifies that actor y's state S2 is a cause for the abandonment of G1 by actor x. The link between x's loyalty goal G2 and the act of x to obey y, denotes that the OBEY(x,y) act is a

precondition for the satisfaction of G2. While the content of particular goals of different episodes within the above SAP is different, as evidenced by Events 4 and 5, the interactions that occur between the goals in these situations are similar. They all involve an actor's desire to satisfy a primary goal G1, as well as a secondary goal, G2 which is ACHIEVE-LOYALTY to y. G1 is in danger of failing because of belief states S1 and S2 of y. Thus, Events 4 and 5 can be handled through the use of similar strategies for dealing with the particular goal and state interaction. The following two figures represent graphically the causal similarity between Events 4 and 5.

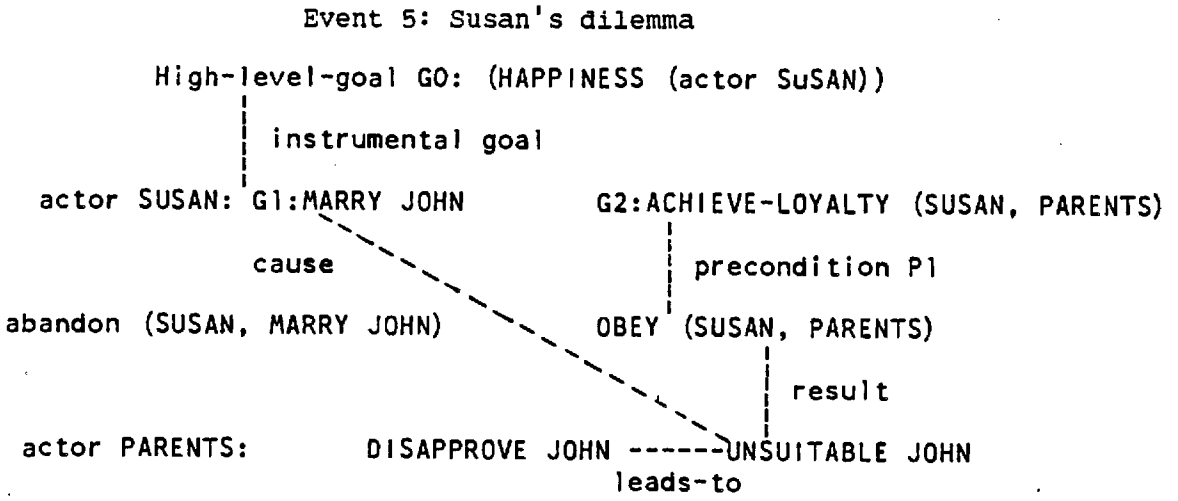


Figure 2

Susan's marriage to John is an instrumental goal to her hapiness. Susan also has as another goal being loyal to her parents. Her parents' disaproval of John's way of living makes them to want her to abandon her plans to marry him. Hence Susan's dilemma: if she wants to fulfill her loyalty goal, she has to abandon her goal to marry John. If she marries John, she has violated the precondition of obeying her parents, thus not fulfilling her loyalty goal.

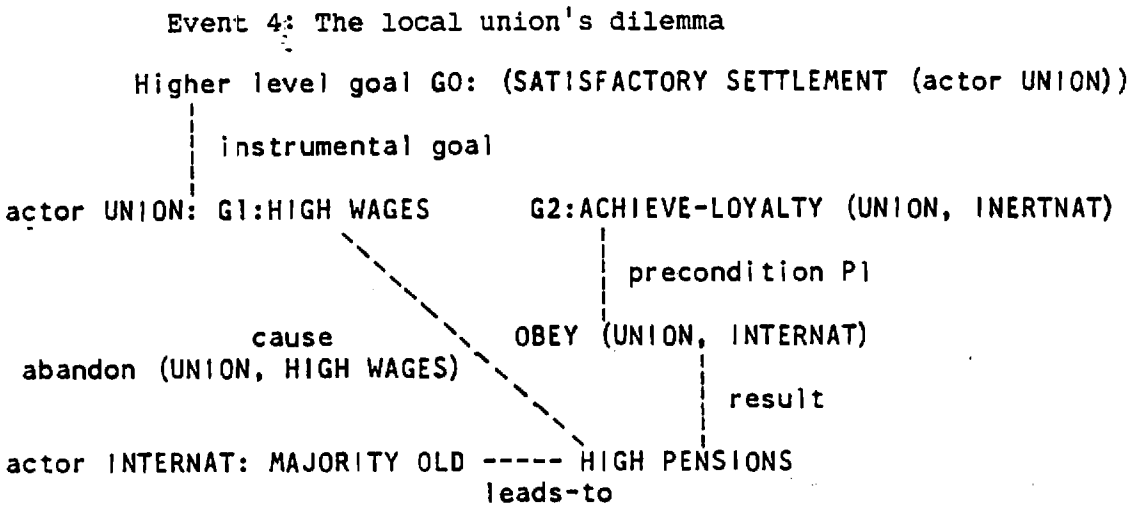


Figure 3

The international union's belief that the majority of its members are old, makes it want all its locals to place the greatest emphasis on high pensions, thus conflicting with the particular local's higest priority on wages. If the local union obeys the international, it will have to abandon its goal for high wages (since it is unlikely that both high wages and high pensions can be simultaneously achieved).

1.4.1. Using SAPs to guide problem solving

One advantage of using SAPs in problem solving is that they contain general planning strategies. These strategies incorporate planning knowledge that depends only on the causal structure of the particular SAP and is independent of domain features. General strategies suggest actions for the problem solver to take with respect to goals depending on the goals' enablement conditions. The general strategies are refined into particular ones depending on the situation (e.g., whether a precondition is necessary for achievement of a goal or not). These, in turn, are used to suggest plans that could be used in the current situation. Thus, a planner would not have to consider all possible plans for the achievement of a goal, but only the ones suggested by the strategies in the SAP. In addition, what is learned in one planning situation is used in planning for a subsequent situation that fits the same SAP.

Let us illustrate the above points using the SAP MISPLACED-LOYALTY. The possible

general strategies include:

1. Fulfill both G1 and G2
2. Abandon G2
3. Partially fulfill G2
4. Abandon G1
5. Partially fulfill G1

To achieve both goals G1 and G2 of x, for example, more specific strategies are (a) plan against state S1 and, (b) plan against state S2. More specialized strategies under the "Plan against S1" strategy include: (i) cause y to change his belief state S1, (ii) wait for y to change his belief state S1.Plans under (i) would indicate, for example, what kind of arguments x can use to change S1. An example of an argument to change S1 in Event 5 might be that John's political views are not so undesirable since politician z's (whom Susan's parents respect) son also has them and z does not seem to disapprove of him.

One other interesting feature of SAPs is that the interactions of the goals of the agents with the goals of the problem solver serve as meta-planning knowledge. The problem solver will suggest one strategy over another depending on how his goals interact with those of the disputants. For example, whether the strategy "Abandon G2" will be suggested in the SAP MISPLACED-LOYALTY depends on whether the ACHIEVE-LOYALTY goal G2 interferes with G3, a goal of the third party problem solver (in conflict resolution G3 is finding an acceptable compromise). In Event 4, the union's high priority to pensions interferes with the mediator's goal of finding a mutually acceptable settlement. If the union abandons its loyalty goal to the international and lowers the priority it attaches to pensions, then the search for an acceptable solution will be facilitated. Hence, the plan "assert own priorities" that is instrumental to strategy "Abandon G2" will be suggested by the mediator.

To use the interaction of the problem solver's goals with those of the agents as meta-planning knowledge, an explicit representation of the problem solver's goals is needed. This capability gives a problem solver a way of extending its vocabulary to include solutions that effect *partial goal satisfaction*. Such a vocabulary is absent from planning work now. The assumption of existing planning systems is that the planning goals should be totally satisfied.

This restricts the range of plans that a problem solver is considering as appropriate in a given situation. If, however, the problem solver had an explicit representation of its goals, it could have the required flexibility to extend the range of plans it is willing to consider. This could be helpful when, for example, there was no plan that could be used for total goal satisfaction.

To fulfill G1 partially, for example, strategy "fulfill goal states subsumed by G1" can be used. In Susan's case, a plan indexed under this strategy could be to continue seeing John without marrying him (since marriage subsumes goals such as companionship and sexual fulfillment). Another strategy to fulfill G1 partially is "act with secrecy". Susan could marry John secretly. This is considered a partial fulfillment of the marriage goal since the social recognition of the marriage relationship would be denied. Which plan is suggested depends on the circumstances. For example, if Susan is pregnant and wants the child, the suggested plan would be to marry John.

The process that generates the justification for a suggested course of action is guided by the abstract causal structure of a SAP. To illustrate this process, consider the SAP MISPLACED-LOYALTY, and suppose that the problem solver suggests to actor x to "Abandon goal G2". If an actor y has another actor x's interests at heart, he must share x's higher level goals and do everything to see them succeed. Thus, if y wants x to abandon a goal that is instrumental to the achievement of a higher level goal of x's, y does not have x's interests at heart. In other words a contradiction is detected. The problem solver points out this contradiction to actor x as justification for the suggestion to x to abandon his loyalty goal G2 towards y.

In order for a problem solver to see whether this justification can indeed be used, he follows the causal link from S2, (see Figure 1) the resulting belief state of y, and checks to see whether this link causes abandonment of a goal of x that is instrumental to a thematic higher level goal of x.

1.4.2. Indexing in SAPs

SAPs are not only processing structures used in problem solving and planning, but they

are also memory structures. SAPs organize generalized episodes (Kolodner et al., 1985) or MOPs (Schank, 1982) as well as single cases. SAPs, like TOPs (Schank, 1982), use domain-independent planning features as indices. Only when a generalized or simple episode is reached, do domain features come into play as indices for accessing a particular case and the associated specific plan. Thus, SAPs are more powerful mechanisms for reminding than generalized episodes since they span different contexts.

Once a solution has been discovered for a particular problem, and the case has been appropriately indexed inside a SAP, it is available through reminding when another case sharing the same abstract goal/plan/state/condition interrelations is being processed. This reminding focuses the attention of the problem solver to a solution that might be directly applicable in the current problem.

To index episodes and planning advice under SAPs, the indices have to be such that they afford efficient retrieval of appropriate strategies and plans regardless of the particulars of the individual goals. One way to do this is to index plans (strategies) under the preconditions they satisfy. In this way, appropriate plans can be returned when preconditions of a situation are known. Another way to index is by effects that need to be achieved. Indexing plans (strategies) under their effects gives a way to ensure that a plan appropriate to the goals to be achieved is chosen. Under the general strategies are indexed the more specific strategies. Under those, the applicable plans are indexed. Under the plans either generalized or simple episodes can be indexed.

The following figure shows part of the indexing structure for SAP MISPLACED-LOYALTY. The upper part of this figure is the same as Figure 1 and depicts the causal structure of MISPLACED-LOYALTY. In addition to the goals and states the preconditions for the accomplishment of the goals are depicted.

Indexing structure for SAP MISPLACED-LOYALTY

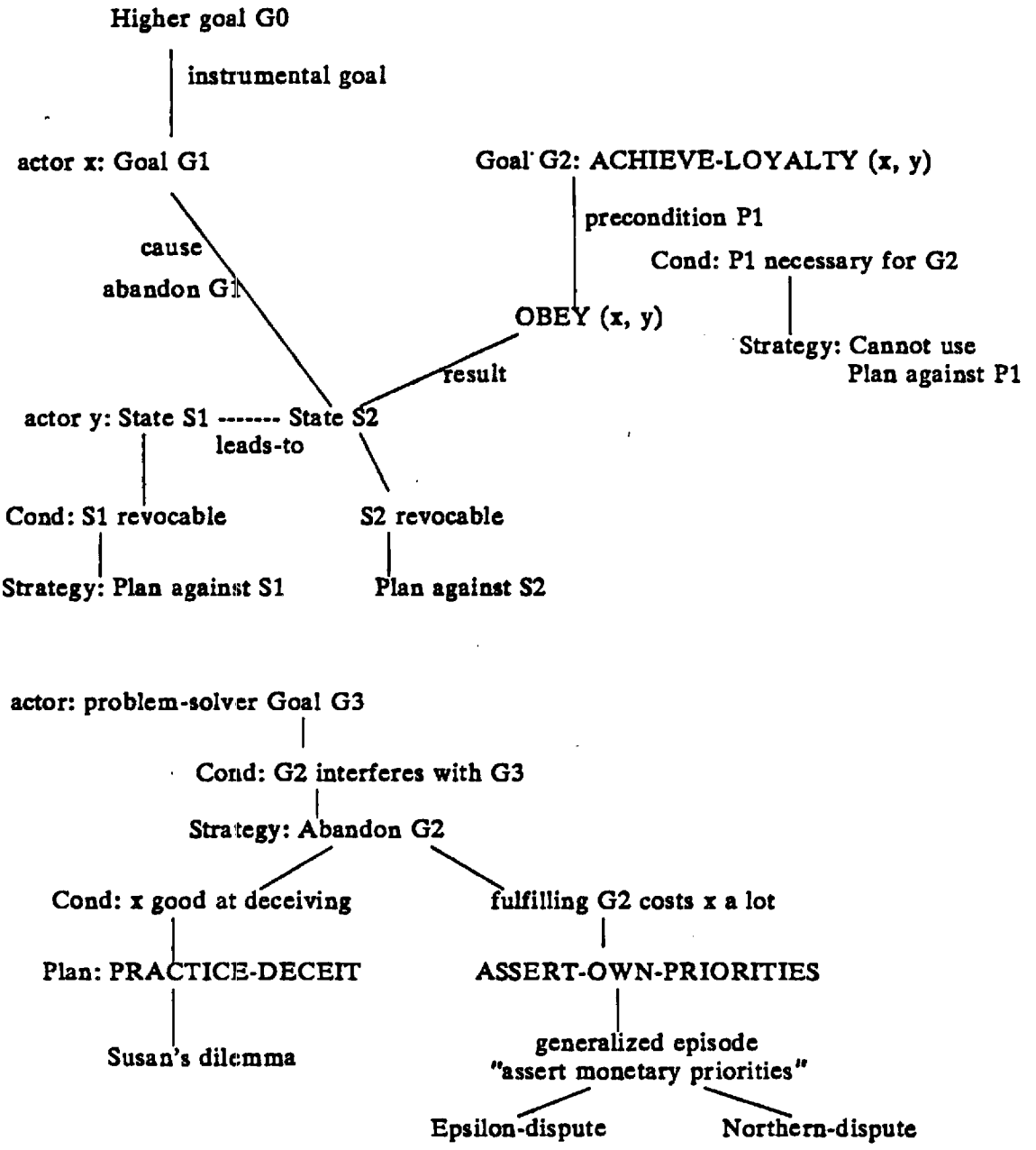


Figure 4

At the leaves of a SAP are pointers to generalized and individual episodes. If a generalized episode is reached, additional indices are traversed to access a particular experience within the generalized episode. For example, the two cases shown organized under the generalized episode "assert monetary priorities" are the Epsilon dispute and Northern dispute. These

disputes are labor mediation cases where the unions involved have asserted their own priorities. Both disputes are indexed in this location in memory since they used the same plan.

1.5. SAPs versus other high level knowledge structures

SAPs are a specialization of Schank's TOPs (Schank, 1982) and apply to situations involving many interacting agents with conflicting goals. While Hammond's TOPs (1986) generally involve goal interactions associated with a *single agent*, SAPs involve interactions of goals in a *multiagent* situation. The inclusion of partial goal fulfillment strategies is another novel feature of SAPs, and one that further differentiates SAPs from planning TOPs (Hammond, 1986), and from Wilensky's (1983) work on planning.

SAPs, unlike TAUs (Dyer, 1983) record the *reason* for the failure of the outcome of a situation. Specifying explicitly the reason for a failure guides *Outcome-Driven Reminding* (Schank, 1982) whose task is to find a memory that will help in solving a current problem. Because SAPs include blame attribution information, they provide a problem solver with *predictions* about the agents' subsequent behavior, something that neither TAUs nor TOPs allow. For example, in SAP MISMANAGEMENT, the mediator can be fairly sure that the company will indeed accept her suggestion of giving stock options to the employees, since the company knows that once it becomes apparent to the employees that it was the management's fault that led it to the brink of bankruptsy, they will not accept a solution that makes the employees the only ones to be penalized.*

1.6. Summary

In this paper I have presented a class of abstract knowledge structures, called SAPs, which represent the causal structure of atypical problem solving situations involving many interacting agents. SAPs perform the following major functions in problem solving:

1. Provide guidance to the problem solver to come up with the appropriate solution

* This has been borne out in real situations, as for example the recent (1985) Eastern Airlines settlement with the International Pilots' Association.

2. Act as a source of preventive and recovery advice to the agents

3. Provide the problem solver with justifications for the proposed solution

4. Organize situations in terms of abstract planning features, thus providing remindings across different domains

SAPs are recognized in terms of expectation violations and blame attribution.

References

Dyer, M.G. *In-Depth Understanding: A Computer Model of Integrated Processing for Narrative Comprehension*, The MIT Press, Cambridge, Mass., 1983.

Hammond, K. "CHEF: A Model of Case-based Planning", *AAAI-86*, pp. 267-271, Philadelphia, Pa., 1986.

Kolodner, J.L. *Retrieval and Organizational Strategies in Conceptual Memory: A Computer Model*, Laurence Erlbaum, Hillsdale, N.J., 1984.

Kolodner, J.L. Simpson, R.L., and Sycara-Cyranski, K. "A Process Model of Case-Based Reasoning in Problem Solving", *IJCAI-85*, pp. 284-290, Los Angeles, Ca. 1985.

Schank, R.C. *Dynamic Memory: A Theory of Reminding and Learning in Computers and People*, Cambridge University Press, Cambridge, Mass., 1982.

Sycara-Cyranski, K. "Arguments of Persuasion in Labour Mediation", *IJCAI-85*, vol. 1, pp. 294-296, Los Angeles, Ca. 1985.

Sycara, K. "Adversarial Reasoning in Conflict Resolution", Ph.D. Thesis, School of ICS, Georgia Institute of Technology, Atlanta, Ga., 1987.

Wilensky, R. *Planning and Understanding: A Computational Approach to Human Reasoning*, Addison-Wesley Publishing Company, Reading, Mass., 1983.

Utility Theory in Conflict Resolution*

Katia Sycara

School of Information and Computer Science
Georgia Institute of Technology
Atlanta, Georgia 30332
Telephone: (404) 894-5550

ABSTRACT

In this paper multi-attribute utility theory is extended to accomodate adversarial problem solving situations involving multiple interacting agents. Such situations are resolved by partial goal satisfaction and persuasion, and have only scantily been described in the AI literature. Utility theory is shown to provide a computational framework to (a) generate a compromise solution that partially satisfies the conflicting goals of the agents, (b) evaluate whether a solution is an improvement on a previously rejected one, and (c) determine the effectiveness of persuasive arguments. Our examples are taken from the domain of labor mediation and are implemented in a computer program, called the PERSUADER.

February 23, 1987

* This work was supported by ARO Grant No. DAAG 29-85-K-0023.

Utility Theory in Conflict Resolution

Katia Sycara

School of Information and Computer Science
Georgia Institute of Technology
Atlanta, Georgia 30332
Telephone: (404) 894-5550

1. Introduction

Perusal of any major newspaper shows that the world is full of conflicts , for example community disputes, labor-management negotiations, international conflicts. Conflicts usually involve two or more active agents, each with multiple goals. For example,

> Event 1: The Redhound company is in contract negotiations with its union. The union's demands are a 20 percent wage increase, a 6 percent increase in pensions and seniority as the sole determining factor for promotions and layoffs. The Redhound company's initial position is a 3 percent wage increase, a 1 percent increase in pensions and the criteria for promotions and layoffs to be determined solely by the company. The parties refuse to move from their positions and as a strike deadline approaches, a mediator is called in.
>
> With the mediator's help, the parties shift their positions to the following: the union's demands become 12 percent wage increase, 4 percent increase in pensions; promotions and layoffs to be governed by seniority and ability as co-determinants with higher weight given to seniority. The company's position becomes 6 percent wage increase; 2 percent increase in pensions; the criteria for promotions and layoffs to be determined solely by the company.
>
> After some more discussions, the parties agree to the following proposal of the mediator: 8 percent wage increase, 4 percent increase in pensions; promotions and layoffs to be governed by seniority and ability with ability receiving a higher weight.

This example illustrates four features of conflict resolution:

1. **There is more than one interacting agent.** In the example, the interacting agents are the mediator, the union and the company.*

2. **Each of the active agents has more than one goal.** These goals are not only different but they interact in certain ways. In Event 1, the union's and company's goals are in conflict in the sense that the union will try to get as much as it can whereas the company will try to give as little as possible.

3. **The agents modify their position during the course of the interaction.** In Event 1

* For simplicity, we have presented the union and the company as monolithic entities. In the real world factions within the union and company may act as different agents.

for example, the union starts with a wage demand of 20 percent increase and after a while it lowers its demand to 12 percent and finally agrees to 8 percent.

4. **The parties agree to a settlement that gives each party something less than their initial goals but more than what the other party would be willing to give.** In Event 1 for example, the union does not get a 6 percent increase in pensions, but gets something better than the 3 percent increase that the company wanted to give it originally.

The example illustrates a resolution of the conflicting goals of multiple interacting agents. Virtually all AI research dealing with multi-agent interactions has assumed that the agents have common or non-conflicting goals [Cammarata83], [Corkill83], [Davis83], [Georgeff84]. People must plan their everyday course of activity, however, taking into consideration the goals and plans of others, which might be in conflict or in concord to their own. The work has thus focused on how these agents can best help each other in achieving their common ends.

There are many situations where the agents have conflicting goals for which compromise solutions would be beneficial. In an automated factory, for example, robots might vie for the use of limited resources. An automated battle manager might be required to coordinate a schedule with another battle manager, while preserving its priorities and goals. A single agent may need to plan for resolution among conflicting goals of its "clients". Such an example would be a job shop scheduler [Fox82] that tries to schedule orders among available machines in the best manner. Even in situations where all agents are assumed to have a common goal, sub-goal conflicts may arise. For example, in a distributed problem solving environment, the machines involved in solving a single problem might face conflicts over use of various computational resources. As research in distributed computing progresses, such situations will soon become very common. Thus, research on conflict resolution in multi-agent situations will become increasingly important.

The AI work up to date on conflict resolution of multi-agent goals (e.g., [Genesereth84], [Rosenschein85]) has modeled the problem as one where the agents arrive at a compromise solution through negotiations using game theory. As has been pointed out, [Stevens63], [Bartos74], game theory is not particularly well-suited to model such situations. The structure of a game is represented by the *payoff* matrix -the set of outcomes (payoffs) associated with the various strategies represented by the various rows and columns of the matrix. Each strategy represents a complete sequence of choices appropriate for a particular sequence of contingencies (the opponent's order of choices). If the number of choices available at each move is more than just a few -a most likely case in any realistic situation, the number of strategies implied (rows and columns of the payoff matrix) is enormous. This, in turn, makes actual solution of the game impracticable. Another drawback is the assumption of game theory that each player knows the whole payoff matrix, namely not only his own but also his opponents payoffs. This is clearly not realistic in conflict situations. The game theory formulation cannot accommodate tactics of persuasion and bluff, which are an integral part of negotiations. It assumes, instead that the payoff matrix remains invariant throughout the game [Luce57].

We have chosen to model the conflict resolution as performed by a third impartial agent, the mediator. Mediation has proved its worth in resolving difficult real world conflicts that the agents themselves were unable to resolve through negotiations. In the non-human environment, the role of the mediator is played, for instance by the scheduler of a job shop orders, or by the co-ordinator in a distributed computing environment. There is no single formalism that has been used to model mediation. Our approach involves the use of past cases as well as heuristics and the use of utility theory as the underlying formalism for portraying the parties' preferences. In this paper, we will concentrate on the uses of utility theory in our mediation model. Our domain of application is labor mediation and our theory is embodied in a computer program called the PERSUADER. The PERSUADER has two general problem solving tasks (a) to construct and propose appropriate compromise

settlements to the parties in a labor dispute and (b) to convince the parties to accept a proposed settlement.

2. What is hard in conflict resolution?

Consider the formidable task that confronts a decision maker who has to propose a resolution that takes into consideration the tradeoffs and preferences of multiple agents with multiple conflicting goals. She has to somehow find a settlement that includes "suitable" values of each issue on which both parties will agree. For continuum-valued issues, the choices that such a decision maker has are infinite. For example, the choices that the mediator had in Event 1 were the combination of the infinity of settlements in the range of 3 to 20 percent increase for wages and 1 to 6 percent increase in pensions initially, and afterwards the 6 to 12 percent range for wages and 2 to 4 percent increase in pensions, as well as the range of differences in seniority language. A blind trial and search process is obviously hopeless.

This difficulty arises in every conflict resolution situation. One way for a problem solver to address this difficulty is to subdivide the range of values for each attribute in a sufficiently large number of pieces and consider only the settlements that result from the finite combination of these values. For example, if one subdivides the wage and pension ranges in Event 1 in 6 pieces, the corresponding resulting values in the set of alternatives would be 6, 7, 8, 9, 10, 11, 12 percent increase for wages and 2, 2.33, 2.66, 2.99, 3.32, 3.65, 3.98 for pensions. The finite set of alternative settlements (considering only wages and pensions) is the set formed by all the combinations of the above values. The (mathematically) optimum settlement may not be a member of this set. This is not, however, a serious drawback for problems dealing with human affairs, since people are not optimizers.

Another difficulty is that usually the agents' goals are expressed non-numerically. How could a problem solver compare such goal values? In Event 1, one of the goals is the achievement of a certain language for seniority. A mediator can say that the language "the criteria for promotions and layoffs are to be determined solely by the company" is weaker seniority language than what the union proposed, but she cannot characterize numerically the magnitude of the difference. One solution to this problem is to adopt an arbitrary numerical scale to characterize the non-numerical attributes. This is the solution we have adopted in our implementation for non-economic issues. The chosen scale goes from 0 to 10. In this scale, for example, 10 denotes the strongest seniority language and 0 the weakest.

If a problem solver is to succeed in finding compromise solutions in situations involving many decision makers, he has to have some method of making inferences about the ways the decision makers evaluate alternative solutions and make choices. The method used by the problem solver should also allow him to take into consideration possible tradeoffs that the decision makers would be willing to accept. Utility theory provides such a methodology. Utility theory is the theory that models the process through which a decision maker evaluates a set of alternatives, so that he can choose the best one. It has also been used in aiding a decision maker to structure his problem in such a way that evaluation of the alternatives is easily accomplished [Whit74], [Keeney75]. In this paper, we concentrate on the novel ways that utility theory can be exploited in problem solving. In our model, utility theory is used by the problem solver to (1) generate potentially acceptable solutions to be proposed to the parties, (2) measure the quality of a modification to a rejected settlement, and (3) determine the effectiveness of persuasive argumentation.

3. Utility Theory in brief*

The concept of *utility* is the basis for selecting among future alternatives and for evaluating past actions. Each time a house is bought, or the choice of a job has to be made, or any

* For an extended treatment of utility theory see [Keeney76].

other form of action has to be taken, some form of assessment of utility of the various alternatives for the decision maker is used in order to make the decision. Each alternative is evaluated in terms of a number of attributes that the decision maker considers important. In buying a house, for example, some of the relevant attributes are cost, distance from work, safety of neighborhood, quietness. Each prospective house is evaluated on each one of these attributes. It is vary rare indeed, that a particular alternative will have the best rating on all the attributes under consideration. Thus, a decision maker must have a way of comparing alternatives with varying attribute values, in order to pick the one that offers him the maximum overall utility, or satisfaction. This is not easy, since the individual utilities associated with the attributes are not linear in general. For example, suppose that the safety of a neighborhood was rated on a scale of 0 (totally unsafe) to 10 (totally safe) points. Further suppose that a mother of small children were to be asked about the utility of this attribute on an (arbitrary) scale from 0 to 100% satisfaction. It is very plausible that she would give the following ratings: zero percent satisfaction for safety in the range of 0 to 4 points, 25% for a safety rating of 5, 75% for a rating of 9 and 100% for a rating of 10. This is obviously a non-linear relationship. Moreover, a different decision maker, a Mafia tough for instance, if asked to rate his satisfaction with the safety attribute on a 0 to 100% scale, would give quite different ratings than the mother. Thus, not only is the utility associated with an attribute not linear, but it also varies with the decision maker.

Another difficulty that arises in comparing alternatives is that a decision maker must accept lower values on some attributes in order to get higher values on others. In other words, he must make trade-offs. Because the measurement scales of the attributes are in general incommensurate, one unit of one attribute does not have the same utility as one unit of another attribute. To continue with the house buying example, even if quietness is measured on a scale of 0 (totally noisy) to 10 (totally quiet), one unit of safety is probably not equivalent to one unit of quietness. In other words, the mother decision maker would not be indifferent (i.e. derive the same satisfaction) between two houses with the same cost, same distance from work but one with 5 units of safety and 4 of quietness and the other with 5 units of quietness and 4 of safety. Thus, a decision maker must know how many units of one attribute he is willing to give up in order to gain one unit of another attribute. The individual utility relations as well as the tradeoff values for the various attributes constitute the *preference structure* of a decision maker. This preference structure potentially varies with each decision maker. Utility theory provides a methodology through which a decision maker's preference structure can be identified, so that utility assessments of alternatives can be made.

We briefly describe formally the general problem that utility theory addresses. It is the following: how should a decision maker decide to choose an act β out of a set A of action options, such that he will be happiest with the consequence/payoff associated with this choice? We give an abstract formulation of this problem. The action options are $A_1,\ldots,A_m$. There are a set of attributes of concern $X_1,\ldots,X_j,\ldots,X_n$, and each option β can be evaluated on each of these attributes to get n indices of value $X_1(\beta),\ldots X_n(\beta)$. In labor mediation, for example, the action options are the various possible contract settlements and the attributes are the contract issues, e.g., wages, seniority, pensions. Let the evaluation of option A_i on attribute X_j be given by the number x_{ij} for $i=1,2,\ldots,m$ and $j=1,2,\ldots n$. Thus, option A_i can be identified with a vector consequence $x_j = (x_{i1},x_{i2},\ldots,x_{ij},\ldots,x_{in})$. Thus, a comparison between two options involves comparisons between two n-tuples. For example, a contract γ with a 47 cents increase in wages and a 6 cents increase in pension benefits would be expressed in the above notation as $\gamma = (47, 6)$, assuming that the ordering of the attributes is (wages, pensions).

Since attributes X_i and X_j may in general be measured in different units, it is meaningless to compare elements x_i and x_j $(i \neq j)$ of an n-valued payoff. Thus, for each option β, we would want to find an index that combines the n-valued payoff $X_1(\beta),X_2(\beta),\ldots.X_n(\beta)$ into a *scalar* value-function v that expresses the preferability of option β for the decision maker. This function is called *value* or *utility function*. Given v, the decision maker's problem is to choose β in A such that v is maximized. In the case where the chosen action has to satisfy

the goals of more than one decision maker, as in labor mediation, the alternative that maximizes the combined payoff of the decision makers is selected.

The utility function v, defined on the consequence space has the property that

$$v(x_1,\ldots,x_n) \geq v(x'_1,\ldots,x'_n) \text{ iff } (x_1,\ldots,x_n) \geq (x'_1,\ldots,x'_n) \quad (1)$$

where the symbol $\geq$ reads "preferred or indifferent to", and "iff" means "if and only if".

It would tremendously simplify the calculations, if we could find a function, call it f, with a simple form such that

$$v(x_1,x_2,\ldots,x_n) = f[v_1(x_1),v_2(x_2),\ldots,v_n(x_n)], \quad (2)$$

where v_i designates a value/utility function over the single attribute X_i. We are interested in conditions when expression (2) holds. The simplest and most useful form that expression (2) can take is the *additive* form, namely

$$v(x_1,\ldots,x_n) = w_1v_1(x_1) + w_2v_2(x_2) + \cdots + w_nv_n(x_n) \quad (3)$$

where w_i designates the weight/importance that a decision maker attaches to each attribute.

One nice property that an additive function has is that it is compensatory in the sense that an increase in the utility of one attribute can compensate for a decrease in the utility of any other attribute. Thus, such a function models well the tradeoffs that a decision maker is willing to consider. Additive functions are the ones most frequently used in practice [Johnson77], [Keeney76].

3.1. Deriving Utilities

If a decision maker has available the utility functions associated with each attribute under consideration, he can take their weighted sum to arrive at an overall utility function for all attributes of interest (cf equ. (3)). This function maps the individual utility values associated with a particular alternative to a single number, the satisfaction of the decision maker with that alternative. To make his final choice, the decision maker selects the alternative that maximizes the overall utility function. As we saw in the previous section, because of the nonlinearity of the relation between the individual utilities and the associated attributes, and the non-commensurability of attribute scales, the assessment of a decision maker's utilities is not an easy problem. To obtain the utility curves of the parties, a problem solver can (a) follow an assessment procedure that elicits the decision maker's utilities via direct questioning, (b) retrieve the utility curves of similar parties from past successful problem solving episodes or (c) hypothesize the shape of the curves from knowledge of domain-specific factors.

There is a variety of utility assessment procedures that directly question the decision maker. Each procedure indicates the kind of questions that can be asked of the decision maker to elicit individual utilities and some guidance as to how the individual utilities are to be combined to obtain the overall utility function. Space limitations prevent us from presenting such procedures, especially since they are well documented in the decision analysis literature (for a survey of these procedures, see [Johnson77]). Though these utility assessment procedures seem to elicit accurate utilities, they are time consuming and in many cases impractical (e.g., they presuppose trust on the part of the decision maker towards the questioner).

Retrieval from memory of the utility curves of similar parties is another way to obtain them, and is the preferred method in our model. In our implementation, the curves are stored as part of the profile frames of the agents whose goals are in conflict. We assume that the utility curves of similar agents for a particular attribute will have the same functional form. This assumption is supported by various experimental studies (e.g., [Swalm66], [Spetzler68]). But what makes agents similar? The answer depends on the domain under investigation. The criteria for similarity of disputants in the domain of labor mediation that the PERSUADER uses are similarity of industry, similarity of geographical location, same

international union. These criteria are reasonable because they reflect the economic realities of the negotiation situation. For example, two paper mills in Georgia are assumed to have the same utility curves for the same contract issues.

Hypothesizing the utility curves of the parties is a third general method that a problem solver can use to obtain utilities. This method relies heavily on domain-specific heuristics. In labor mediation, the factors that are used in the heuristics are the state of the economy in the industry, the unemployment rate for the bargain unit's job classification in the area, and the structure of the bargaining unit (e.g., proportion of skilled vs. unskilled workers, young vs. old). The following figure shows how the factor of economic boom or recession impacts the health-benefits curve of a union.

POSSIBLE UNION UTILITY CURVES FOR HEALTH BENEFITS

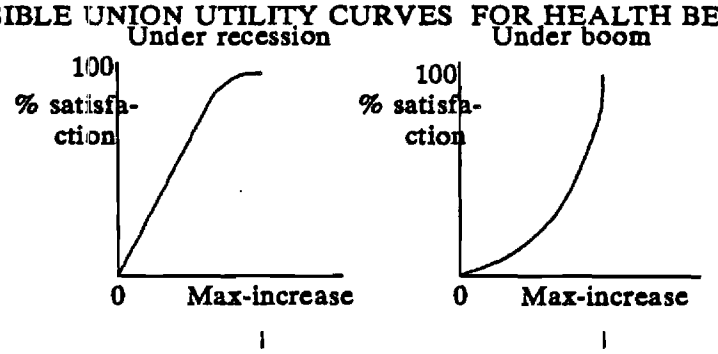


Figure 1

Notice that in both cases the union will not be satisfied at all if it is not given any health benefits increases, and it will be 100% satisfied if it is given the maximum increase. Under boom, the union will be less than 50% satisfied if it is given an increase of magnitude $[\frac{1}{2}Max-increase]$, whereas under recession, it will be more than 50% satisfied if the conceded increase is $[\frac{1}{2}Max-increase]$. Thus, the two curves in the figure reflect qualitatively the realities of a union's satisfaction under two different economic conditions. It is reasonable, therefore for the mediator to hypothesize the shape of these curves. Elementary calculus gives analytic expressions for these two curves. The utility curve under recession can be expressed as

$$y(x) = \frac{-100}{Max^2}(x-Max)^2 + 100$$

and the utility curve under boom can be expressed as

$$y(x) = \frac{100}{Max^2}x^2$$

where, for notational simplicity, Max-increase is denoted by Max.

Hypothesized utility curves and those derived from the utility curves of similar parties are not as accurate as the ones derived from direct assessment techniques. This, however, is not a great disadvantage in our model, since these curves are used to propose an initial

solution and get modified in the course of problem solving via persuasive argumentation.

The PERSUADER derives the utility curves of a decision maker by retrieving the curves of similar decision makers that have been encountered in previous problem solving episodes. If no such past experience is available, the PERSUADER hypothesizes the utility curves by using a set of domain-specific heuristics to select the appropriate curves from a number of curves that it knows about. Once the utility curves have been obtained for the contract issues under negotiation in the present case by either method, the PERSUADER constructs the overall utility function for each party by asking the parties directly for the weights they attach to these issues and forming the weighted sum (cf eq. 3) of the retrieved curves. The PERSUADER uses the utility curves of the parties constructed thus in three ways during the problem solving process: (1) to suggest a potentially acceptable compromise solution, (2) to evaluate the quality of the modification to a rejected settlement, and (3) to determine the effectiveness of persuasive argumentation.

3.2. Utility theory in generating a compromise solution

As has been illustrated in section 2, the task of proposing a potentially acceptable resolution that takes into consideration the tradeoffs and preferences of multiple agents with multiple conflicting goals is a hard one. How can utility theory help? A utility function models a decision maker's preferences so that he can evaluate a set of multi-attribute alternatives and select the best one. In conflict resolution situations the alternative that would be the most preferable for one agent would most likely be the least preferable for another, since their goals are in conflict. Thus, a third party problem solver is faced with the problem of how to select a compromise solution that will be potentially acceptable to all parties. We assume that the parties are reasonable enough to know that they cannot get the settlement that is most preferable to them, since somebody else is bound to object to it. We have considered two heuristic criteria that seem reasonable and can guide the problem solver in selecting the "best" compromise solution: (1) maximizing the *joint payoff**, and (2) minimizing the *payoff difference* of the parties.

The solution that maximizes the joint payoff is objectively the "best" settlement that can be achieved taking into consideration the parties' subjective utilities. In addition, it is necessary that the proposed solution be perceived as equitable to obtain acceptance. Minimizing the payoff difference is an intuitive expression of the concept of equity.

To apply the first criterion, one can proceed as follows: By range subdivision of each of the attribute values and combination of the resulting values, (see section 2), a finite set of alternatives is constructed. For simplicity, let us consider the case where there are two issues under consideration. For example, in Event 1, considering only wages and pensions as the issues, one of the alternatives would be (6, 2), namely a 6% increase in wages and 2% increase in pensions (assuming the range subdivision of section 2). Adapting eq. (3) for two issues, the company's (and union's) utility curves can be expressed by the general formula

$$v(x_1,x_2) = \alpha v_1(x_1)+(1-\alpha)v_2(x_2), \tag{4}$$

where α and $(1-\alpha)$ are the weights and $v_1(x_1)$ and $v_2(x_2)$ the utility curves for wages and pensions for each respective party. Thus, $v(6,2)$ can be calculated for each party. The joint payoff of the parties is given by the general formula

$$U(x_1,x_2) = u_1(x_1,x_2) + u_2(x_1,x_2), \tag{5}$$

where $u_1(x_1,x_2)$ is the company's utility curve and $u_2(x_1,x_2)$ is the union's utility curve for settlement (x_1,x_2). The joint payoff of the parties for each settlement under consideration can be calculated using eq. (5). Then, the alternative that gives the maximum of these values is selected and proposed.

* Maximizing the joint payoff has been suggested in [Raiffa82].

Another possible criterion could be to select the most fair solution, namely the one with the smallest difference in the parties' payoffs. This is done as follows: Once the parties payoffs for each alternative have been calculated using eq. (4), the difference

$$U_d(x_1,x_2) = \left| u_1(x_1,x_2) - u_2(x_1,x_2) \right| \tag{6}$$

is calculated (assuming u_1 is the utility curve of the company and u_2 of the union). The alternative that minimizes this difference is selected.

In order to decide which criterion the PERSUADER would use, we ran a few examples using each one. Maximizing the joint payoff very often gave contracts with quite unequal utilities for the parties. In those experiments, the payoff of one party would be so low as to practically guarantee rejection of the settlement by that party. On the other hand, minimizing the difference can lead to absurd results. For instance, this criterion would not be able to differentiate between alternatives one of which gives both parties a payoff of 40, and another that gives both parties payoff 70 (since in both cases the payoff difference is 0). Hence, we chose to combine the two criteria and select the alternative that minimizes the difference and maximizes the joint payoff. This is done by computing the joint payoff (eq. 5) and the payoff difference (eq. 6) for each alternative, taking the difference of these two and selecting the alternative that maximizes this difference.

In our implementation, we assume that a solution that affords both parties a payoff greater than or equal to 70%* will be accepted by the parties. If the solution under consideration gives both parties such a payoff, it is proposed without further evaluation. The parties might of course choose to accept a settlement that gives less than 70% payoff. Usually, conflicts requiring mediation involve tough compromises that give both parties payoffs that are less than 70%. Because of the parties' understandable reluctance to accept low payoffs, methods for fine-tuning rejected solutions are needed. We have identified two methods that can be used to obtain compromise: (a) improvement of rejected solutions, and (b) change in the perceived payoff of a rejected solution. The following figure depicts the top level control for generating compromise solutions.

* This number has been checked for approximate accuracy by practicing mediators. The reason that 100% satisfaction with a resolution is not necessary is that the parties are assumed to be reasonable, in the sense that they know that they need to compromise.

GENERATING COMPROMISE SOLUTIONS

1. Generate compromise solution s1 that minimizes the difference and maximizes the joint payoff

2. If s1 gives both parties payoff greater than 70%, assume s1 accepted. SUCCESS.

3. Otherwise, propose s1

4. If s1 accepted by both parties, then SUCCESS.

5. Otherwise, while persuasive arguments can be retrieved, attempt to persuade rejecting party, p1.

6. If s1 accepted by p1, then SUCCESS.

7. Otherwise, modify s1 to s1' that is more favorable to p1.

8. If s1' improves the payoff of p1 more than it penalizes p2, then go to step 3.

9. Otherwise, go to step 7.

3.3. Utility theory in finding a better solution

No matter how the utility functions of the parties have been obtained, they are not completely accurate [Johnson77], [Shepard64]. Moreover, the preference structures of the agents can change during problem solving [Bartos74], [Swingle70]. Thus, a suggested compromise solution may be rejected by either or both parties. A problem solver needs to be able to suggest another solution that will be no worse than the rejected one in the sense that it will have at least the same chance of been accepted. To do this, a problem solver has to have some criterion that progress is being made every time a new solution is to be proposed. The parties' payoffs give such a criterion. A problem solver employs various plans to create a new solution by modifying the rejected one [Sycara85c]. These plans are domain dependent and the result of their application on the rejected solution is predictable. For example, if there is a high turnover of workers in a company, a mediator can infer that they would not be very interested in strong seniority language, and thus she can employ a plan to further weaken the seniority language in a situation where the company has rejected a suggested settlement.

To see how the parties' payoff can be used as a criterion of whether a modified settlement has improved its chance of acceptability, consider the following example: Suppose that a proposed contract with 40 cents increase in wages and 10 cents increase in pensions and with payoffs 52% for the company and 62% for the union, is rejected by the company. The mediator proposes a 3 cents pension reduction resulting in the contract (40, 7). The mediator calculates the payoffs of the parties for the contract (40,7) by using eq. (4). Suppose these payoffs are 61% for the company (an increase of 8%) and 58% for the union (a decrease of 4%). The criterion that is used to decide whether to suggest the modified contract is that it increase the rejecting party's payoff by a greater amount than it might decrease the payoff of the party that had accepted the previously proposed contract.* In the above example, the

* In most conflict situations if a resolution that was acceptable to one party is modified in favor of the opposing party, the resulting resolution will give the party that had accepted a smaller payoff than the previous resolution.

contract (40, 7) will be suggested. Thus, a problem solver does not waste time in proposing solutions that are inferior to rejected ones. The incremental solution improvement process is akin to *hill climbing* and the above criterion affords the test for proceeding.

3.4. Utility theory in persuasion

In conflict resolution, a problem solver uses persuasive argumentation to convince a party that rejected a settlement to accept it, or to narrow the parties' differences with respect to the issues' values by convincing a party to accept a lower value than the one he demanded. To accomplish this, a problem solver needs a computational handle on the notion of "convincing" someone to accept a settlement/value that was previously unacceptable to him. The notion of payoff supplies such a handle. "Convincing" somebody can be modeled as increasing the payoff that the settlement/value gives him. Hence, the task of a persuader can be viewed as finding the most effective argument that will increase a party's payoff with respect to a settlement/value.**

Equation (3) shows that there are two ways to increase a party's payoff: (a) by changing the weight/importance the party attaches to an issue, and (b) by changing the value of an issue. These two ways can be viewed as a persuader's argumentation goals. In the rest of this section we give a brief description of how a utility-derived formulation can guide a persuader in the selection of argumentation strategies and particular arguments to achieve the argumentation goals (for a more detailed treatment, see [Sycara85a, Sycara85b]).

For simplicity, let us consider the case where there are two issues under consideration. Then, eq. (3) becomes

$$v(x_1,x_2) = \alpha v_1(x_1) + (1-\alpha)v_2(x_2) \tag{7}$$

where x_1 and x_2 are the values of the two contract issues, v_1 and v_2 are the individual utility curves associated with the issues, and α and $(1-\alpha)$ are the weights (or relative importance) that the party accords the contract issues. If the weights are changed, the payoff also changes. This reflects the intuitive notion that satisfaction with a thing is a function not only of the intrinsic value of the thing but also of the importance that we attach to it, our view of it. The sign of the partial derivative of v with respect to α indicates the direction of change of v.

$$\frac{\partial v}{\partial \alpha} = v_1 - v_2; \text{ so if } v_1 \geq v_2, \text{ to increase } v \text{ increase } \alpha \tag{8a}$$

$$\text{if } v_1 < v_2, \text{ to increase } v \text{ decrease } \alpha. \tag{8b}$$

Thus, relations (8a) and (8b) show how the weights can be changed, so that an increase in payoff will result. Moreover, (8a) and (8b) show that the change in the weights of one party can be carried out independently of any weight changes for the other party. Since there are more than one issues involved, a persuader needs to find out (a) which issue's importance she should try to change, and (b) in what direction (increase or decrease). Equations (8a) and (8b) give us a criterion for answering these two questions. A persuader has access to the parties' utility curves as well as the importance that the parties attach to the various issues. Thus, when a party has rejected a proposed settlement, a persuader can check the relationship ($\geq$ or $<$) of the utility curves for the values of the issues in the proposed resolution. Then, she can use equation (8a) or (8b) to decide which issue's importance to increase or decrease. This is the procedure the PERSUADER uses. In the PERSUADER, arguments are accessed with respect to (a) the issue to which they pertain and (b) whether they increase or decrease the issue's importance [Sycara85c].

Another argumentation goal of a persuader is to change the assessment of the value of

** Modeling a persuader's task thus, can be applied to general situations of persuasion, not just conflict resolution. In the general case, a persuader needs to convince somebody of a proposition, not just a settlement.

the issue under discussion in the persuadee's eyes. In the mediation domain, "the issue under discussion" is a contract issue, and its value is the monetary value of the contract issue. In the utility theory model, changing a party's assessment of the value of an issue is equivalent to changing the party's satisfaction curve at that value, thus raising the party's payoff. Consider, for instance the situation where a company assesses an increase of 20 cents an hour in wages as "too high". In the utility theory formulation, this can be translated as $vc(20)=35$ (i.e., the satisfaction that the company would derive if it were to give a wage increase of 20 cents per hour would be 35%). Convincing the company that this increase is not so high, is equivalent to raising its satisfaction (or in mathematical notation, $vc(20)>35$).

The PERSUADER uses two methods to generate persuasive arguments: (1) recall of arguments that have proven effective in past similar circumstances, and (2) construction of arguments using knowledge of the adversaries' goals. The strategy to select persuasive arguments for presentation to the disputants is:

1. Access arguments that have proven effective in similar circumstances, or construct a novel argument if none are retrieved

2. Give preference to arguments that change the importance of the issue under consideration

Arguments that change the importance of an issue are preferred over those that change the utility curve of a persuadee because the resulting change in the perception of the issue is easier to predict.

It would be an ineffective persuader, however, if one were unable to recognize when argumentation is useless. This happens when a party's payoff is so low that no amount of persuasion can convince him to change his position. Since a persuader can calculate the parties' payoffs that a proposed settlement gives them, she can decide whether argumentation is effective. In our implementation, we consider that if a proposed settlement gives a party a payoff less than 50%*, it is useless to argue and it is better to try to form another potentially acceptable settlement.

4. An example from the PERSUADER

The PERSUADER is a computer program that, presented with an impasse in labor negotiations between a company and a union, discovers and proposes reasonable contract settlements and persuades the parties to accept the proposed settlement. It uses *precedent-based reasoning* [Sycara85c], namely the retrieval of similar contracts from its memory of past experiences, to use as a starting point for its reasoning. We present the following example, where the PERSUADER resolves a collective bargaining dispute involving the VARTA transit company and the bus drivers' union. The user's feedback is in boldface. The current impasse in negotiations is presented. The union wants a 12% wage increase and a 6% increase in pensions, while the company proposes a 3% increase in wages and no pension increase.

(intro VARTA-dispute)
The present contract negotiation is VARTA company contract negotiations
The negotiations are at an impasse, which I will try to resolve
Looking to propose a contract that is acceptable
to #<M-LOCAL 16271553> (VARTA union)
and #<M-TRANSIT-COMPANY 16274557> (VARTA company)
The impasse resulted from the following:
VARTA-union has made the demands

* We have checked this number with practicing mediators who told us that it is about right.

#<M-WAGE-GOAL 16272354> (12% wage increase)
#<M-PENSION-GOAL 16272565> (6% increase in pensions)
VARTA company has responded with the proposals
#<M-WAGE-GOAL 16272377> (3% wage increase)
#<M-PENSION-GOAL 16272612> (no increase in pensions)

Searching memory for current competitors' contracts...
Having found current competitors' contracts...
The contract I am considering is the current contract
for #<M-TRANSIT-COMPANY 16271555> (NARTA company)

The PERSUADER checks the retrieved contract to see whether wages and pensions, the issues in the current dispute were negotiated in the case of the NARTA company. Indeed, they were. The frame representing the NARTA company contains its utility curves for wages and pensions. The frame representing NARTA's union correspondingly contains its utility curves for wages and pensions. NARTA company's wage utility curve is (in algebraic representation) $w_c(x_1) = \frac{100}{(B_1 - A_1)^2}(x_1 - B_1)^2$, where A_1 and B_1 are the variables representing the endpoints of the range for wage values. NARTA company's pension utility curve is $p_c(x_2) = \frac{100}{(A_2 - B_2)}(x_2 - B_2)$, where A_2 and B_2 are the variables representing the endpoints of the range for the pension values. Correspondingly, the NARTA union's utility curve for wages is $w_u(x_1) = \frac{-100}{(B_1 - A_1)^2}(x_1 - B_1)^2 + 100$, where A_1 and B_1 have the same meaning as above, and its pension utility curve is $p_u(x_2) = \frac{100}{(B_2 - A_2)}(x_2 - A_2)$, where A_2 and B_2 have the same meaning as above.

Subdividing the range [12, 3] of #<M-WAGE-GOAL 16322757>
into 4 pieces...
Subdividing the range [6, 0] of #<M-PENSION-GOAL 16323031>
into 4 pieces...
The considered contracts are formed by combining
all the endpoint values derived from the range subdivision.
There are two issues in dispute...
Combine the values of these two issues.
There are 25 contracts under consideration.

Need to find the weights the parties attach to the issues.

*** THE MEDIATOR TALKS PRIVATELY TO #<M-LOCAL 16271553> ***
What would you #<M-LOCAL 16271553> say is the relative importance
on a scale of 0 to 1 of #<M-PENSION-GOAL 16323031>
with respect to #<M-WAGE-GOAL 16322757> for you? .7

*** THE MEDIATOR TALKS PRIVATELY TO #<M-TRANSIT-COMPANY 16274557> ***
What would you #<M-TRANSIT-COMPANY 16274557>
say is the relative importance
on a scale of 0 to 1 of #<M-PENSION-GOAL 16323031>
with respect to #<M-WAGE-GOAL 16322757> for you? .4

Using company #<M-TRANSIT-COMPANY 16271555> (NARTA company)
utility curves as curves of #<M-TRANSIT-COMPANY 16274557>
Using #<M-LOCAL 16273551> (NARTA union)
utility curves as curves of #<M-LOCAL 16271553>

Calculating company's utility for each of the 25 contracts
using weighted sum of company utility curves for the issues.
Calculating union's utility for each of the 25 contracts
using weighted sum of union utility curves for the issues.

Checking to see whether any contract gives both parties payoff > 70...
No contract gives both parties payoff > 70.

Calculating joint payoff for each of the 25 contracts
using simple sum of parties' payoffs.
Calculating difference payoff for each of the 25 contracts
using simple difference of parties' payoffs.

Calculating difference between the joint payoff and the difference payoff
for each of the 25 contracts.
Selecting contract (6.0, 4.0) corresponding to maximum difference
since this contract minimizes the difference and maximizes the joint payoff.

The contract #<M-CONTRACT 16276047>
with 6.0 change in the value of #<M-WAGE-GOAL 16322757>
and 4.0 change in the value of #<M-PENSION-GOAL 16323031>
minimizes the difference and maximizes the joint payoff
Contract #<M-CONTRACT 16276047>
will be proposed to solve the impasse.

The proposed contract has a 6% wage increase and a 4% increase in pension benefits. Though the PERSUADER calculated that the company cannot quite afford this economic package, it is nevertheless proposed since the PERSUADER knows of a plan to pass the extra cost to the consumer. The PERSUADER proposes this settlement to both sides.

*** THE MEDIATOR TALKS PRIVATELY TO THE PARTIES ***

Do you #<M-LOCAL 16271553> accept #<M-CONTRACT 16276047>
as a way to solve the impasse ? yes

Do you #<M-TRANSIT-COMPANY 16274557> accept #<M-CONTRACT 16276047>
as a way to solve the impasse ? no

The company rejects the proposed contract. The PERSUADER's goal now is to convince the disagreeing party to agree. It finds, however that the rejecting party's payoff for this contract is 45% and, considering it too low decides not to try to use persuasive arguments to increase it. It now considers another plan for an acceptable settlement. First, the PERSUADER checks to see whether the plan's preconditions are satisfied. Then it checks to see whether the plan's application will result in an improved solution.

Looking at the plan called "try to reduce the cost of pensions"

With respect to pensions,
since the bargain unit consists mainly of young workers
a reduction in pension cost seems acceptable

The settlement has still 6% wage increase but only 1% increase in pension benefits. The PERSUADER checks the parties' payoffs. The company's payoff increased by 20% (from 45 to 65) and the union's payoff decreased by 12% (from 67 to 55). So, the new settlement is proposed.

The contract #<M-CONTRACT 16276047> which resulted
from the plan #<M-REDUCE-PENSIONS 16324663>
will be proposed to solve the impasse.

*** THE MEDIATOR TALKS PRIVATELY TO THE PARTIES ***

Do you #<M-LOCAL 16271553> accept #<M-CONTRACT 16276047>
as a way to solve the impasse ? **no**

Do you #<M-TRANSIT-COMPANY 16274557> accept #<M-CONTRACT 16276047>
as a way to solve the impasse ? **yes**

*** THE MEDIATOR'S SOLILOQUY ***

Since the #<M-LOCAL 16271553> rejected the contract
I need to find the weights #<M-LOCAL 16271553>
attaches to the issues

The PERSUADER asks for the weights of the rejecting party again to check whether they have changed. In this case, the union's weights have not changed. Hence, the payoff remains at 55%.

The #<M-LOCAL 16271553> 's payoff for the contract #<M-CONTRACT 16276047>
with 1.0 change in the value of #<M-PENSION-GOAL 16323031>
and 6.0 change in the value of #<M-WAGE-GOAL 16322757>
is (55.0)
Since #<M-LOCAL 16271553> rejected the contract,
the payoff needs to be increased, if appropriate

Since the value in utility curve of #<M-WAGE-GOAL 16322757>
is greater than the value in utility curve for #<M-PENSION-GOAL 16323031>
try to find an argument to increase the weight of wages
or, equivalently, decrease the weight of pensions.

The PERSUADER observes that it is unusual for a union with a majority of young members to give such high importance to pensions and forms the hypothesis that the great weight to pension increases may be due to a goal set by the international union. The program checks and verifies this hypothesis in the current case.

Then, the PERSUADER searches memory for appropriate arguments, namely arguments that have been used in the past to convince the rejecting party (the union) that the importance it attaches to pensions is too high and it evaluates whether the argument would be applicable in the present case.

*** THE MEDIATOR TALKS PRIVATELY TO #<M-LOCAL 16271553> ***

It is o.k. for the international union
to have a high pension goal,
but your workers are mostly young
so, they won't be disappointed
to receive lower pension benefits.

Do you agree ? yes

Do you #<M-LOCAL 16271553> accept #<M-CONTRACT 16276047>
as a way to solve the impasse ? yes

The mediator accepts congratulations
---

5. Summary

In this paper, we have presented how utility theory can be incorporated in problem solving in situations involving multiple agents with multiple conflicting goals. Utility theory is used for (a) generation of solutions to be proposed to the parties, (b) measuring the quality of a modification to a rejected solution, and (c) measuring the effectiveness of persuasive argumentation.

References

[Bartos74] Bartos, O., *Process and Outcome of Negotiations*, Columbia University Press, New York and London, 1974.

[Cammarata83] Cammarata, S., McArthur, D., and Steeb, R., "Strategies of cooperation in distributed problem solving", *IJCAI-83*, pp. 767-770, Karlsruhe, West Germany, 1983.

[Corkill83] Corkill, D. D., and Lesser, V. R., "The use of meta-level control for coordination in a distributed problem solving network", *IJCAI-83*, pp. 748-756, Karlsruhe, West Germany, 1983.

[Davis83] Davis, R., and Smith R. G., "Negotiation as a metaphor for distributed problem solving", *Artificial Intelligence*, vol. 20, pp. 63-100, 1983.

[Fox82] Fox, M. S., Allen, B., and Strohm., G. "Job-Shop scheduling: An investigation in Constraint-Directed Reasoning", *AAAI-82*, pp. 155-158, Pittsburg, PN., August 1982.

[Genesereth84] Genesereth, M. R., Ginsberg, M. L., and Rosenschein, J. S., "Cooperation without Communication", *HPP Report 84-36*, 1984.

[Georgeff84] Georgeff, M. A., "Theory of action for multi-agent planning", *AAAI-84*, pp. 121-125, Austin, Texas, 1984.

[Johnson77] Johnson, E. M., and Huber, G. P., "The Technology of Utility Assessment", *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-7, no. 5, pp. 311-325, May 1977.

[Keeney75] Keeney, R. L., and Nair, K., "Decision analysis for the siting of nuclear power plants-The relevance of multiattribute utility theory", *Proceedings of the IEEE*, vol. 63, pp. 494-500, 1975.

[Keeney76] Keeney, R. L., and Raiffa, H., *Decisions with Multiple Objectives*, John Wiley and Sons, New York, 1976.

[Luce57] Luce, R. D., and Raiffa, H., *Games and Decisions: Introduction and Critical Survey*, John Wiley and Sons, New York, 1957.

[Raiffa82] Raiffa, H., *The Art and Science of Negotiation*, The Harvard University Press, Cambridge, Mass., 1982.

[Rosenschein85] Rosenschein, J. S., and Genesereth, M. R., "Deals Among Rational Agents", *IJCAI-85*, vol. 1, pp. 91-99, Los Angeles, Ca., August 1985.

[Shepard64] Shepard, R. N., "On subjectively optimal selection among multiattributed alternatives", *Human Judgement and Optimality*, Shelley, M. W., and Bryan, G. L., Eds. Wiley and Sons, New York, 1964.

[Spetzler68] Spetzler, C. S., "The development of a corporate risk policy for capital investment decisions", *IEEE Transactions on Systems, Science and Cybernetics*, SSC-4, pp.

279-300, 1968.

[Stevens63] Stevens, C. M., *Strategy and Collective Bargaining Negotiation*, McGraw-Hill Book Company Inc., 1963.

[Swalm66] Swalm, R. O. "Utility theory-insights into risk taking", *Harvard Business Review*, vol. 44, pp. 123-136, 1966

[Swingle70] Swingle, P., Ed. *The Structure of Conflict*, Academic Press, New York, 1970.

[Sycara85a] Sycara-Cyranski, K. "Arguments of Persuasion in Labour Mediation", *IJCAI-85*, vol. 1, pp. 294-296, Los Angeles, Ca., 1985.

[Sycara85b] Sycara-Cyranski, K. "Persuasive Argumentation in Resolution of Collective Bargaining Impasses", *Proceedings of the Seventh Annual Conference of The Cognitive Science Society*, pp. 356-360., Irvine, Ca., 1985

[Sycara85c] Sycara-Cyranski, K. "Precedent-based Reasoning in Expert Labor Mediation", *Georgia Institute of Technology Report Git-ICS-85/22*, Atlanta, Ga., 1985

[Whit74] Whitmore, G. A., and Cavadias, G. S., "Experimental determination of community preferences for water quality-cost alternatives", *Decision Sciences*, vol. 5, pp. 614-631, 1974.

Paper submitted to AAAI-87 February, 1987.

Capitalizing on Failure Through Case-Based Inference*

Janet L. Kolodner
School of Information and Computer Science
Georgia Institute of Technology
Atlanta, Georgia 30332

1. Introduction

Over the past several years, there has begun to be a great deal of interest in case-based and analogical reasoning (e.g., Alterman, 1986, Ashley, 1986, Carbonell, 1983, 1986, Hammond, 1986, Holyoak, 1984, Kolodner, et al., 1984, 1985, Rissland, 1986, Simpson, 1985). Case-based reasoning is a problem solving method in which previous reasoning experiences are used directly to solve a new problem, rather than solving the problem from scratch using generalized methods. The major advantages of a case-based approach are that it can provide shortcuts in problem solving and that it can help a reasoner avoid repeating previously-made mistakes.

We shall see that previous failures serve several purposes during problem solving. They can provide warnings of the potential for failure in the current case, and they may also provide suggestions of what to do instead. Analyzing the potential for failure in a new case, a necessary part of capitalizing on an old failure, may require the problem solver to gather additional information, thus causing the problem solver to change its focus of attention. A previous failed case that was finally solved correctly can help the problem solver to change its point of view in interpreting a situation if that is what is necessary to avoid potential failure.

We shall illustrate the processes involved in capitalizing on failure using examples from two domains: common-sense mediation of everyday disputes and menu planning. Case-based resolution of common-sense disputes is implemented in the MEDIATOR (Kolodner, et al., 1985, Simpson, 1985), an early case-based reasoning program. JULIA (Cullingford & Kolodner, 1986) interactively solves problems in the catering domain. The processes that capitalize on failure are implemented in JULIA.

* This work is supported in part by NSF under Grant No. IST-8317711 and Grant No. IST-8608362, by ARO under Contract No. DAAG29-85-K-0023, and by ARI under Contract No. MDA-903-86-C-173. Programming of the examples, and much work on analogical reasoning that is incorporated into JULIA's case-based reasoner was provided by Hong Shinn. Discussions with other members of the AI Group, past and present, have also been useful.

2. Background

In the simplest case, making a case-based inference involves the following steps:*

1. Recall a relevant case from memory
2. Determine which parts of that case are appropriate to make the necessary problem solving decision for the new case (i.e., focus on appropriate parts of the previous case)
3. Achieve the targetted problem solving goal for the new case by making an inference based on the old case
4. Check the consistency of what is derived in step 3 to the new case

Consider, for example, the following case:

Avocado Dispute 1

> A problem solver is attempting to resolve a dispute over possession of an avocado. Two people want it. The problem solver is attempting to fill in the underlying goals of the disputants (i.e., why does each want the avocado?). It is reminded of a dispute in which two kids wanted the same candy bar. They both wanted to eat the candy bar, and the reasoner compromised by dividing the candy bar equally between them, having one divide it and the other choose his half first.

The problem solver has already been reminded of another case (step 1). Because the problem solver's goal is to infer the underlying goals of the disputants in the avocado case, it focuses on the underlying goals of the disputants in the candy dispute (step 2). They both had the goal of eating the whole candy bar. This goal was inferred through a *default-use* inference. The reasoner makes the case-based inference that the disputants in the avocado dispute also want to eat the disputed object (i.e., the avocado) (step 3). Because this hypothesis is consistent with what is already known about the case (step 4), the representation of the case is updated to include this inferred knowledge.

When a recalled case resulted in failure, however, reasoning is not as straightforward. Consider, for example, the following:

Avocado Dispute 2

> A problem solver is attempting to resolve a dispute over possession of an avocado. Two people both want it. The problem solver is trying to infer the underlying goals of the disputants. This time it is reminded of a case where two sisters both wanted the same orange. The problem solver in that case inferred the sisters' goals by using a *default-use* inference to infer that both disputants wanted to eat the orange. It turned out, however, that the goal of one of the disputants was to use the peel of the orange to bake a cake. The *default-use* inference applied to the orange as a whole led to selection of the wrong plan for resolution of the conflict, and the plan

* Each of these steps, of course, is a complicated process. For more information about step 1, see Kolodner (1983, 1984), Hammond (1986), Holyoak (1984), Schank (1982); about step 2, see Kolodner, et al. (1985), Simpson (1985); for step 3, see Alterman (1986), Ashley (1988), Carbonell (1983, 1986), Hammond (1986), Kolodner (1985, 1986), Kolodner et al., (1985), Rissland (1986), Simpson (1985); for step 4, see Simpson (1985).

failed. We shall call this part of the case *orange-dispute-f*.

The problem solver reinterpreted the dispute and solved it. The goals of the sisters were amended: one wanted possession of the fruit of the orange, the other its peel. Their underlying goals were also amended: one wanted to satisfy hunger by eating the fruit, the other wanted to bake with the peel. It finally resolved the problem by dividing the orange in a better way. One sister was given the fruit and the other was given the peel. We shall call this part of the case *orange-dispute-s*.

The problem solver also analyzed its failure in *orange-dispute-f*, and added its analysis to its memory of that case: Failure was due to a *wrong-goal inference*. *Default use* applied to the entire disputed object (orange) resulted in failure, while *default use* applied to parts of the orange (the peel and the fruit) would have resulted in success.

Suppose now that the problem solver is reminded of *orange-dispute-f*, the case that resulted in failure. This case acts as a warning to the problem solver of the potential to make a faulty inference in the current case. It must check to see if the inference used previously would also result in error in the current case. The question that must be asked of the avocado dispute based on analysis of the orange dispute is whether an avacado also has parts used for different purposes that might predict the goals of the current disputants better than if they were computed by applying *default-use* to the whole avacado. In other words, based on its reminding of *orange-dispute-f*, which failed, case-based reasoning alerts the reasoner to the fact that if the disputed object has several parts, the goals of the disputants may have something to do with the parts and not necessarily with the avocado as a whole.* The potential for failure is flagged and two alternative solutions are presented.

Errors in reasoning can happen during any problem solving step. The problem might have been misunderstood initially, resulting in incorrect classification of the problem or incorrect inferences during the problem elaboration phase. Since problem understanding is an early part of the problem solving cycle, such misunderstandings and incorrect inferences propagate through to the planning phase, resulting in a poor plan. A problem might be understood correctly and all the necessary details known about it, but might still be solved incorrectly because poor decisions were made while planning a solution. In general, such errors are due to faulty problem solving knowledge. The problem solver might not have complete knowledge, for example, about under what circumstances a particular planning policy or plan step is appropriate. Finally, a problem might be solved correctly but carried out incorrectly by the agent carrying out the plan, or unexpected circumstances might cause execution to fail. Reminding of a case where any of these things happened warns the the problem

* It may be judged in this case that inference based on the parts is inappropriate (since one rarely plants avocado seeds).

solver of the potential for the same type of error in the new case. If the previous case was finally resolved correctly, details of its correct resolution suggest correct decisions for the current case.

3. Some Problem Solving Assumptions

Before presenting the set of processes that capitalizes on previously-failed cases, we briefly present the relevant parts of our problem solving paradigm. First, when we refer to problem solving, we include the entire cycle of understanding a problem and elaborating its features, coming up with a plan for its solution, executing that plan, analyzing the results, and if necessary, going back to the beginning and trying again. Our own previous work (Kolodner, et al., 1985, Simpson, 1985) and that of others (e.g., Hammond, 1986) has shown that case-based inference can be used for a variety of tasks during any of these problem solving phases.

The second important assumption of our paradigm is that memory access and problem solving are happening in parallel (Kolodner, 1985, Kolodner & Cullingford, 1986). The memory's job is to integrate the case that is currently being reasoned about into the memory that already exists (Schank, 1982), resulting in remindings. Memory can return generalized knowledge (e.g., knowledge structures or rules) for the problem solver to use or a previous case that is similar to what the problem solver is currently dealing with. As the problem and its solution are further elaborated, memory is able to recall both more relevant general knowledge and better related cases for the problem solver to use.

Our third important assumption is that case-based reasoning is happening in the context of a set of reasoning goals and that, in addition to the case-based reasoner, other reasoners are also keeping track of those goals and making any suggestions they can (Kolodner, 1987). Thus, in addition to the case-based reasoner, a problem reduction problem solver might be available to break the problem into smaller parts, while a constraint propagator might do forward chaining inferences, and a truth maintenance system might be checking for inconsistencies and constraint violations. Something we'll call the overall problem solver keeps track of reasoning goals and subgoals as they come up, and each of the reasoners watches the goal network and attempts to achieve any goal it can.

Finally, the processes we present below assume that reminding has been of the failed part of a case that might have been resolved correctly later. In the case of the orange dispute, for example, we assume reminding has been of the episode that failed, *orange-dispute-f*. Reminding during problem solving may be of either the successful or the failed

version of any case. When reminding is of the successful instance of solving it, the faulty reasoning that preceeded the successful solution is bypassed and a good solution is suggested immediately. The problem solver is never alerted to possible problems. Only when reminding is of a failed attempt at resolving a case is the problem solver alerted and the analysis described below done.

4. The Process

Given this set of assumptions, we see that the problem solver might be reminded of a previous case that resulted in failure any time during problem solving. Because of this, the processes that capitalize on previous failures must be applicable during any part of the problem solving cycle. The following set of steps are executed any time during problem solving that a failed case is recalled.

1. Determine whether the failed case was ever followed up on, and, if so, recall the entire reasoning sequence that followed it.

This step makes alternatives that were attempted previously to solve the recalled problem available to the problem solver.

In the representation we are currently using, each full analysis of a problem is kept separately with pointers between them. Thus, the representation for a case that failed and was reanalyzed, such as the orange dispute, is actually represented as two cases. The first is the one that failed (*orange-dispute-f*), where one set of assumptions was made about the goals of the disputants. That one includes the mistaken problem description, the suggested plan (cut it in half), feedback after suggesting or carrying out that plan (after suggesting that the orange be cut in half), and the analysis of what went wrong (a *wrong-goal-inference*). The first (failed episode) also includes a pointer to the next problem solving episode, i.e., the reasoning that is carried out to solve the problem after the failures of the first episode have been diagnosed and repaired. Thus, *orange-dispute-f* points to *orange-dispute-s*, where the problem is described as one where the disputants have the second set of goals, and the solution plan that goes with that (divide agreeably) is recorded.

2. Recall or determine what was responsible for the previous failure.

In some instances, responsibility for failure will already have been attributed during previous reasoning. In that case, this step is an easy step of retrieving the error attribution from the representation of the case. In other instances, there might not have been any analysis of why the previous problem occured. When this happens, it is appropriate for the problem solver to try to figure out why the previous error happened. We do not go into that process in this paper.*

* If responsibility for failure is not known at the end of this step, it is still possible to capitalize on the failure.

In general, failures happen because some inference was made incorrectly or not made at all. This might be due to faulty or missing information about the problem itself, or faulty or incomplete problem solving knowledge. An analysis of a failure may record only which inference was made incorrectly or was not made, or it may record the reasons why the inference was made incorrectly. As we shall see, the better an analysis of a previous failure is, the more the problem solver will be able to capitalize on the failure. The best analysis of a failure will record reasons for faulty reasoning all the way back to a point in the reasoning where it could have been corrected, i.e., where the missing or faulty information can be obtained or fixed. For example, failure in *orange-dispute-f* can be traced to a *wrong-goal inference*. The goals were inferred incorrectly. The reason for this is that *default-use* was applied to the wrong object (i.e., to orange as a whole rather than the parts of the orange). The reason for this is that the problem solver was viewing the orange in the wrong way: as a whole rather than as a thing with functional parts. If the reasons for this inference error are recorded to this level, then by using this case and following the set of steps to be presented, the problem solver will be able to consider whether some other object might be better viewed as a thing with functional parts. If only the fact that the goal was inferred incorrectly were recorded, it would not have as much to go on, but would only be able to consider if there is another goal associated with the object.

3. Determine the relationship of the decision currently being focussed on to the previous failure and refocus as required:
   (a) Was the decision analogous to the one the problem solver is currently trying to make responsible for the failure? If so, maintain current problem solving focus.
   (b) If not, was the decision analogous to the one the problem solver is currently trying to make dependent on the one responsible for the failure, or alternatively, did the value the problem solver is currently attempting to derive change in the final solution to the problem? If so, refocus the problem solver on the decision analogous to the one that was responsible for the previous failure.
   (c) If not, then refocus as in (b) to be careful or maintain current focus to be fast.

When the decision the problem solver is currently trying to make was responsible for the previous failure (i.e, the answer to 3(a) is yes), then more effort must go into making that decision. This is the case in *avocado dispute 2*. The problem solver has the goal of inferring the goals of the disputants, and it was this decision that was responsible for the failure in *orange-dispute-f*.

The more interesting cases, however, is when the answer to 3(b) is yes. In these cases, some decision other than the one currently being attempted was responsible for the previous failure. The problem solver will have to *refocus* itself on

that decision, and (re)make it for the current case before continuing. Consider, for example, the following:

Panama Canal Dispute

> Both Panama and the United States want possession of the Panama Canal Zone. The problem solver is attempting to figure out how to classify the dispute. The problem solver is reminded of the dispute between Israel and Egypt over the Sinai. Both wanted the Sinai, and the problem solver had originally classified it as a *physical dispute* over possession of the land. It had therefore suggested that they cut it down the middle and share it. Both Israel and Egypt balked. On further analysis, the failure of this suggestion was tracked down to a set of *missing-goal inferences*. The goals of Israel and Egypt with respect to the Sinai had not been inferred. Israel wanted military control of the area for security reasons, while Egypt wanted possession of the land itself for reasons of national integrety. This interpretation makes the dispute into a *political dispute* rather than a physical one, i.e., one for which political alternatives are suggested rather than alternatives having to do with the physical object itself.

Responsibility for the failure in the previous case (the Sinai Dispute) had already been tracked down to missing goal inferences. The problem solver is currently attempting to decide what kind of dispute it is (e.g., physical or political?). The original classification of the Sinai Dispute as a physical dispute was not per se the reason that solution failed. Rather that decision was based on the goals of the disputants, which had been inferred incorrectly previous to attempting classification. The physical classification, however, changed to political in the final analysis, and was dependent on what was responsible for the failure in reasoning. Remindling of the Sinai Dispute should *refocus* the problem solver on the set of decisions that were responsible for its failure, namely inference of disputant goals.

If the decision being focussed on at the beginning of this set of steps was a correct one for the previous case and if it did not change when the case was reanalyzed (case c), there is no reason why the problem solver must consider the previous failure at all. However, a careful problem solver will also consider whether that failure is possible in the current environment, thus refocusing itself on whatever caused the failure previously before going on.

In cases where the problem solver changes its focus, it continues by trying to redo the task that could have been made in error, following the set of steps below. If the problem solver changes a decision it had made previously, then it must also remake any decisions that depended on it before going on. After this set of steps is complete, the problem solver must refocus appropriately to finish solving the problem. Processing that happens in the course of recomputing already-made decisions may direct the problem solver in different directions than it had been planning when it was interrupted by the failed case. On the other hand, if there are no other recomputations to be made or if no other problem solving directions are suggested, the problem solver continues after this step as it had been planning originally. That is, it goes back to the

goal it was working on when it reached this step and continues from there.

The processing that happens after step 3 depends on whether or not a successful solution was ever found in the previous case and whether or not analysis can be or has been done of the previous failure. If there was neither a solution found to the previous problem nor an explanation of the previous failure, then only an analysis of the potential for failure can be contributed by the the previous case. And, if there is no explanation of the failure, then less can be contributed than if there is an explanation. With an explanation, we know what features of the previous case were responsible for the failure and we can check for the presence of those in the new case. Without that explanation, we can use the justifications for previously made inferences and see if they hold in the new case, but such analysis is in a sense "superstitious" since no causal explanation available.

4. Recall the inference rules and justifying conditions used to infer the focused-on portion of the failed case. IF there was followup, THEN also recall the inference rules and justifying conditions used to infer the focused-on portion of each of the followup cases.*

The inference rules and justifying conditions of any failed cases will be used to check for the potential for failure in the current case. Those from the successfully-resolved case will be used to guide the problem solver to a correct decision.

In the case of *orange-dispute-f*, the inference rule used to infer the goals of the disputants was *default-use* applied to the disputed object. It is justified by its preconditions, i.e., there is an object of current interest (the orange) that has a default use (eating). It might also have been justified by its use previously in the candy dispute, where it worked fine. For *orange-dispute-s*, there were two inference rules used to infer the goals of the disputants. In one case, *default-use* was applied to the fruit of the orange, in the other it was applied to the peel of the orange. The fruit and peel of the orange are its major parts and each are used for different purposes.

5. Check to see if there is the same potential for failure in the new case. This is done by a variety of methods. We list two here.
   (a) Check the reason why the reasoning error was made in the first case. An error can be made because of incomplete information, because of faulty information, because of a faulty inference rule, or because of faulty focus (which might itself be tracked down to one of these causes).
   (b) Determine if the justifying inference rules and conditions from the failed and successful cases also hold in the new case.

Let us consider (a) first. This is the way we determine potential for failure in a new case if we know why the previously-made decision failed. If a previous reasoning error was made because of lack of knowledge, the appropriate knowledge is

* Recall that the problem solver might have refocused its goals in the last step, so the portion of the case being focused on now might not be the one originally considered.

now sought for the current case. If it was because of faulty information, this step will require clarification of the analogous knowledge in the new case. If it was because of a faulty inference rule, that rule will be ruled out in this case. And if it was because of faulty focus (probably due to one of the other types of error), a suggestion will be made from the previous case of where to focus in the new case. Analyzing the orange dispute using this step, we find that the reason for the *wrong-goal-inference* was faulty focus. Focus had been on the orange as a whole while it should have been on its functional parts. The suggestion is thus made to focus on the functional parts of the avacado, rather than the avacado as a whole in inferring the goals of the disputants with respect to the avacado. As in the analysis of the orange dispute from above, in the next steps, the reasoner will either ask the disputants which parts of the avacado they are interested in or will decide that the only functional part that is worth considering is the fruit.

When there is no knowledge about why a previously-made decision was in error, the best that can be done is to evaluate whether conditions that led to that decision are also present in the current case. This is case (b). These conditions can be found in the justifications for the value that was computed previously. If justifications of both the failed and the successful decision are applicable in the new case, an evaluation must be done of which is best. In *orange-dispute-f*, for example, the goals of each disputant were computed using a *default-use* inference applied to the disputed object. Justification for the *default-use* inference comes from its antecedent clause, which asks whether there is some major default use for the object in question that has an "obvious" goal associated with it. An orange and an avocado, of course, both have the same default use (eating) and "obvious" goal (satisfy hunger). In *orange-dispute-s*, the goals of each disputant were computed using a *default-use* inference applied to the functional parts of the disputed object. Justification for this application of this inference rule is a combination of the justification for choosing the objects to be focussed on (the disputed object has functional parts) and the antecedent clause of *default-use* applied to each of those parts.

Using the orange dispute as a model for the avocado dispute, justifications for each of the goal decisions made in resolving that dispute are evaluated with respect to the avocado dispute. Since the avacado has a default use (eating), the inference from *orange-dispute-f* can be made. Since it also has parts with default uses (the fruit is eaten while the seed can be planted), the inferences from *orange-dispute-s* can also be made. In this case, further evaluation is needed to determine which way to make the inference. While case-based reasoning, in this case, does not provide an answer, it does

warn of the potential for misinterpreting the case and it also provides suggestions of alternate interpretations. It thus acts as a *preventive measure* to aid in avoiding failure.

It is interesting to note that the knowledge necessary to do the computations just described may not yet have been considered (e.g., the problem solver may not have considered if an avacado has parts used for different purposes). Sometimes, gathering appropriate knowledge consists of just an easy question to the user. In some cases, however, answering the questions posed in this set of steps may require significant reasoning. This extra computation, while significant, is done only when a previous case points to the need to look out for a problem. As we stated previously, it is a preventive aid in avoiding failure.

The output of this step is an evaluation of whether the previous failure could happen in the new case, and if the previous case was solved successfully, an evaluation of whether the previous successful solution is applicable to the new case. Based on these two evaluations, the reasoning continues.

6.

(a) If the previous failure will not repeat itself in the new case, go on with the problem solving. The (failed) suggestion from the previous case can be transferred to the new case if there is some independent reason that it can be supported or or a decision can be made independent of the recalled case.

(b) If the previous failure could repeat itself in the new case, rule out the inference rule or value used previously for the new case.

(c) If the previous successful solution is judged applicable to the new case, use it and apply case-based reasoning methods to derive a value for the new case based on it.

(d) If the previous successful solution or any of the interim solutions from the previous problem are judged inapplicable to the new problem, rule them out for the new case.

(e) If both the failed and successful solutions to the previous problem are judged applicable to the new one, use some decision-making procedure to decide between them.

5. Case-Based Inference in JULIA

In the following problem solving session, we see JULIA following the set of steps above to capitalize on a previous problem solving failure. JULIA (Cullingford & Kolodner, 1986, Kolodner, 1987) is designed to be an automated colleague whose task is to help a caterer design a meal. JULIA's problem solving components include a case-based reasoner, a problem reduction planner, a constraint propagator, and a reason-maintenance system. It also has a memory for events. Each decision JULIA makes when it is solving a problem is recorded along with the justifications for the decisions. Thus, in later problem solving, those justifications are available to use in case-based reasoning.* In the first case-based reasoning

* See Kolodner (1988) for a description of the representational support for these processes, especially the content and structure of the justifications JULIA maintains. Kolodner (1987) describes the interactions between the different

example, we see JULIA using a previous case to avoid serving a spicy Mexican meal to people who don't eat spicy food. The previously-failed case, which failed because of a lack of this information, causes JULIA to gather the appropriate information in the new case, thus letting it plan for those people immediately. This makes JULIA retract two previously-made decisions (the entree and the cuisine). JULIA refocuses itself on the cuisine, retracts and remakes that decision, and continues with its problem solving.

The dialog begins with JULIA introducing itself and the user stating her problem. We state the problem in English here. Actual input for this problem is a conceptual representation.

```
<JULIA> Hello.  I'm JULIA, a Caterer's Advisor.  May I help you?
USER =>
 I'm having a dinner next week for my research group.
 There is no room for all of them in the dining room.
<JULIA>
*** The initial problem is:
***** Frame #<FRAME 34502032> *****
Name: M-MEAL774
   Isa: (<M-MEAL>)
   Category: INDIVIDUAL
   Slots:
      ACTUAL-RESULT: NIL
      CHARACTERS: (?HOST ?GUESTS ?PARTICIPANTS)
      CONSTRAINTS: (<C-LIMITED-SPACE778>)
      DEFINED-SLOTS: NIL
      DESCRIPTOR: NIL
      EXPECTED-RESULT: NIL
      FOLLOW-UP: NIL
      GOALS: (<E-EAT776> <S-HUNGER777>)
      GUESTS: <*JLK*S-GROUP>
      HOST: <*JLK*>
      ORDER: NIL
      PARTICIPANTS: (?HOST ?GUESTS)
      SETTING: <*JLK*S-HOUSE>
      STEPS: NIL
      TIME: NIL
```

JULIA's problem reduction problem solver starts out. It immediately reduces the instantiate-a-meal task to its component parts and begins working on the first of them, the meal descriptor task. Meal descriptors include, among other things, the meal's cuisine.

```
***** Planning for the task MEAL-TASK775 .....
```

processes.

TASK-TREE = ((<MEAL-TASK775>))

The active goal chain is (M-MEAL)

Reducing the task MEAL-TASK775 to subtasks (<MEAL-DESCRIPTOR-TASK>

<MAIN-COURSE-TASK> <APPETIZER-TASK> <SALAD-TASK> <DESSERT-TASK>)

***** Planning for the task MEAL-DESCRIPTOR-TASK .....

TASK-TREE = ((<MEAL-DESCRIPTOR-TASK> <MAIN-COURSE-TASK> <APPETIZER-TASK>

<SALAD-TASK> <DESSERT-TASK>) (<MEAL-TASK775>))

The active goal chain is (MEAL-DESCRIPTOR <= M-MEAL)

As part of this task, JULIA attempts to formulate a cuisine, but can't do it with the information supplied so far by the user, so it asks.

Constraint Satisfaction .....
Checking constraints for slot CUISINE of MEAL-DESCRIPTOR783 ...
Instantiating slot CUISINE of frame MEAL-DESCRIPTOR783 ...
<JULIA> What cuisine would you like?
USER => mexican
<JULIA>
Constraint Satisfaction .....
Checking constraints for slot CUISINE of MEAL-DESCRIPTOR783 ...
Instantiating slot CUISINE of frame MEAL-DESCRIPTOR783 ...
--> The value is MEXICAN-CUISINE

JULIA goes on to its next task, choosing a main course, and suggests chili for the main course (we leave out the way it did that). At that point, it is reminded of a meal in which chili was served. If this meal had been a successful one, the reminding would provide suggestions about other parts of the meal. In this case, however, some guests, who did not eat hot food, were left unsatisfied.

***** Planning for the task MAIN-COURSE-TASK .....
TASK-TREE = ((<MAIN-COURSE-TASK> <APPETIZER-TASK> <SALAD-TASK>
<DESSERT-TASK>) (<MEAL-TASK775>))
The active goal chain is (SC-MAIN-COURSE <= M-MEAL)
<JULIA> What about CHILI791 for the main course?
USER => ok
<JULIA>
***** Reminded of MEAL80, where chili was the main course
***** Case-based reasoning with the case MEAL80
*** Trying to do analogy-mapping with the case MEAL80 ...
Checking if the previous plan for goals S-HUNGER80 E-EAT80 was successful .....
Previous plan execution failure found

The set of goals failed was S-HUNGER80 E-EAT80

It was because ((NOT EVERY ONE ATE SPICY DISH))

JULIA will try to avoid making this mistake again. It finds (through looking at the representation of the previous case) that the previous failure was because of a missing constraint about spices and seeks to find out if this constraint should be taken into account in the current case. After asking, it finds that to be so and creates a a "non-spicy-food" constraint for the current case. It propagates that constraint and checks it against what it has already decided. It finds out that chili and Mexican food are spicy, and rules both out. Because choosing a main course is dependent on having a value for cuisine, it deletes the choose-a-main-course task from the task network, reschedules the meal-descriptor task and the choose-a-main-course task, and attempts the meal-descriptor task again in an effort to choose a cuisine.

*** Attempting to avoid the previous plan failure.....
The assigned blame was that C-NON-SPICY-PREF80 had not been considered.
To avoid previous plan failure ...
Asking the user of a missing constraint C-NON-SPICY-PREF
<JULIA> Is there anyone who doesn't like spicy food? (How many?)
USER => 3
<JULIA>
Trying to propagate the constraint C-NON-SPICY-PREF793 .....
  -> Generating a new constraint C-NON-SPICY-CUISINE794
  -> Generating a new constraint C-NON-SPICY-DISH795
Applying constraint C-NON-SPICY-DISH795 to CHILI791
    -> Aborting CHILI791
Applying constraint C-NON-SPICY-CUISINE794 to MEXICAN-CUISINE
    -> Aborting MEXICAN-CUISINE
  -> Killing the current task MAIN-COURSE-TASK .....
  -> Rescheduling MEAL-DESCRIPTOR-TASK MAIN-COURSE-TASK into the task network .....
***** Planning for the task MEAL-DESCRIPTOR-TASK .....
 TASK-TREE = ((<MEAL-DESCRIPTOR-TASK> <MAIN-COURSE-TASK> <APPETIZER-TASK>
          <SALAD-TASK> <DESSERT-TASK>) (<MEAL-TASK775>))
 The active goal chain is (MEAL-DESCRIPTOR <= M-MEAL)
Constraint Satisfaction .....
  Checking constraints for slot CUISINE of MEAL-DESCRIPTOR783 ...
  -> Applying constraint C-NON-SPICY-CUISINE794 to slot CUISINE
    -> The slot CUISINE is not yet filled in
  Instantiating slot CUISINE of frame MEAL-DESCRIPTOR783 ...

Because there has been little in the way of preferences offered by the user up to now, JULIA cannot suggest a new cuisine by itself at this point. It asks the user again for a cuisine preference, this time telling the user constraints on the preference. The user suggests Italian, and JULIA goes on. To complete the menu, JULIA continues its reasoning, choosing lasagne for the main course and is reminded of a case in which vegetarians were at a lasagne dinner and could not eat. JULIA knows

that in the previous case, they could have eaten if the meatless version of the dish had been served, and proposes the same in this case. The meal JULIA finally comes up with includes vegetarian antipasto as the appetizer, veggie lasagne and italian bread for the main course, mixed green salad as the salad, and ice cream for dessert.

6. Discussion

In our scheme, potential failures can be encountered and thus need to be dealt with during any step of the problem solving. Any time the problem solver encounters a case with a previous problem, it considers whether there is the potential for that problem in the new case. This may cause it to refocus itself until the potential for failure is determined, and if such potential is determined and the problem solver has to retract decisions made previous to the current one, then it must remake any decisions dependent on those decisions. Such processing, of course, requires that the problem solver be integrated with a reason-maintenance system that keeps track of the dependencies among its decisions. Other steps require that the reasoner record justifications for each of the decisions it makes. We have not done a great deal of work in these areas, but our experience so far leads us to believe that a standard truth maintenance system (Doyle, 1979, McAllester, 1980, DeKleer, 1986) is not adequate to do all of the work we need such a system to do. In particular, in addition to its standard bookkeeping functions, such a system will need strategies or policies to follow in making decisions about how to make the world consistent when a condition check fails, or will need to interact with a reasoner that can make such decisions. While it is standard for a truth maintenance system to retract decisions that are inconsistent and to propagate those retractions as far as it needs to, in the problem solving situation we are looking at, it is often more advantageous to try to satisfy constraints in a different way (e.g., to replace a retracted value with another that satisfies the necessary constraints).

Hammond (1986) takes the complexity out of this issue by having the reasoner explicitly try to avoid mistakes in one of its early planning steps. The advantange of this, of course, is that after potential mistakes are discovered, the problem solver need only keep them in mind during the remainder of problem solving rather than having to deal with new issues and possible change of focus part of the way through. There is thus no need for the complexity of a truth maintenance system. On the other hand, the reasoner can only avoid those mistakes that can be foreseen at the onset of problem solving, but cannot avoid mistakes that the problem solver might not be able to anticipate until late in the problem solving.

Carbonell (1986) deals with this issue in yet another way. His work assumes that each old problem is stored as a sequence of reasoning steps, and that any time two problems are similar in their set of steps, the second is stored on with the first, branching from it at the place they begin to be different. Thus, once the case-based reasoner is reminded of a previous case, it has available to it all of the cases that have been solved by the same initial set of steps as the one it is currently trying to solve. This means that at each decision point in the problem solving, each of the previous decisions that have been made are available along with their justifications. Reasoning similar to that described in this paper happens to evaluate which of the possibilities is appropriate for the new case. The advantages of this method are similar to the advantages in Hammond's method: the problem solver, in general, never needs to refocus itself, and there is no need for a truth maintenance system. The major disadvantage, however, is that once Carbonell's problem solver finds a set of previous cases that are similar to its current one, it is wedded to that set, and no other cases that might be similar along a different set of dimensions can contribute to the problem solving.

7. Summary

Previous problem solving failures can be a powerful aid in helping a problem solver to become better over time. When a previous case in which an error was made is recalled, it flags the potential for a similar mistake and the reasoner considers whether the same potential for error exists in the new case. The direct result of this is that reasoning is directed to that part of the current problem that was responsible for the previous error, sometimes changing the problem solver's focus. Evaluation of the potential for error in the current case may require the problem solver to gather knowledge it doesn't already have, another way focus might be redirected. A case with an error may also suggest a correct solution for the new case. The combination of these helps the problem solver to avoid repeating mistakes and suggests shortcuts in reasoning that avoid the trial and error of previous cases.

8. References

Alterman, R. (1986). An Adaptive Planner. *Proceedings of AAAI-86*, pp. 65-69.

Ashley, K. (1986). Knowing What to Ask Next and Why: Asking Pertinent Questions Using Cases and Hypotheticals. *Proceedings of the Eighth Annual Conference of the Cognitive Science Society.*

Carbonell, J. G. (1983). Learning by Analogy: Formulating and Generalizing Plans from Past Experience. In Michalski, R. S., Carbonell, J. G, & Mitchell, T. M., *Machine Learning*, Tioga.

Carbonell, J. G. (1986). Derivational Analogy: A Theory of Reconstructive Problem Solving and Expertise Acquisition. In Michalski, R. S., Carbonell, J. G., & Mitchell, T. M., *Machine Learning II*, Morgan Kaufmann Publishers, Inc.

Cullingford, R. E. & Kolodner, J. L. (1986). Interactive Advice Giving. In *Proceedings of the 1986 IEEE International Conference on Systems, Man, and Cybernetics.*

de Kleer, J. (1986). An Assumption-Based TMS. *Artificial Intelligence*, vol 28, pp. 127-162.

Doyle, J. A truth maintenance system, *Artificial Intelligence*, vol 12, pp. 231-272.

Hammond, K. (1986). Learning to Anticipate and Avoid Planning Problems through the Explanation of Failures. *Proceedings of IJCAI-86.*

Holyoak, K. J. (1984). The Pragmatics of Analogical Transfer. In Bower, G. (Ed.), *The Psychology of Learning and Motivation*, Academic Press.

Kolodner, J. L. (1983). Reconstructive Memory: A Computer Model. *Cognitive Science*, vol 7.

Kolodner, J. L. (1984). *Retrieval and Organizational Strategies in Conceptual Memory: A Computer Model.* Lawrence Erlbaum Associates.

Kolodner, J. L. (1985). Experiential Processes in Natural Problem Solving. Technical Report No. GIT-ICS-65/23. School of Information and Computer Science. Georgia Institute of Technology. Atlanta, GA 30332.

Kolodner, J. L. (1986). Some Little-Known Complexities of Case-Based Inference. *Proceedings of TICIP-86.*

Kolodner, J. L. (1987). Coordinating Case-Based and From-Scratch Reasoning. Submitted to the Ninth Annual Conference of the Cognitive Science Society.

Kolodner, J. L. & Cullingford, R. E. (1986). Towards a Memory Architecture that Supports Reminding. In *Proceedings of the 1986 Conference of the Cognitive Science Society.*

Kolodner, J. L. & Simpson, R. L. (1984). Experience and Problem Solving: A Framework. *Proceedings of the Sixth Annual Conference of the Cognitive Science Society.*

Kolodner, J. L., Simpson, R. L., & Sycara, K. (1985). A Process Model of Case-Based Reasoning in Problem Solving. In *Proceedings of IJCAI-85.*

McAllister, D. (1980). An Outlook on Truth Maintenance. Artificial Intelligence Laboratory, AIM-551, MIT, Cambridge, MA.

Rissland, E. L. & Collins, R. T. (1986). The Law as a Learning System. *Proceedings of the Eighth Annual Conference of the Cognitive Science Society.*

Schank, R. C. (1982). *Dynamic Memory.* Cambridge University Press.

Simpson, R. L. (1985). A Computer Model of Case-Based Reasoning in Problem Solving: An Investigation in the Domain of Dispute Mediation. Ph.D. Thesis. Technical Report No. GIT-ICS-85/18. School of Information and Computer Science, Georgia Institute of Technology, Atlanta, GA.

> Two sisters are quarrelling over an orange. Their mother surveys the situation and proposes that each sister take half of the orange. One of the sisters complains, since she wants to use the whole peel for baking. Realizing the real nature of the conflict, the mother suggests that the sisters divide the orange agreeably: one will take the fruit and eat it, while the other will take the peel and use it for baking.

Analysis of this example shows that while the mother thought that both sisters had the same goal, she was mistaken. Though their sub-goals were in conflict, their goals were not. Stepping back and considering the real goals rather than the manifest ones resulted in a goal concordance. The following shows how this analysis is transferred in understanding and making a prediction about another situation. Here, we imagine the mother reading the following story in the paper:

> Egypt and Israel both want possession of the Sinai. The US suggests they cut it down the middle. Both Egypt and Israel complain.

Analogy to the orange dispute allows her to conclude that possession of the Sinai is merely a subgoal, that the real goals of the two countries should be considered, and that a mutually-agreeable split based on those goals be sought.

THE MEDIATOR

Our MEDIATOR project [3], [4], [6] resolves common sense disputes based on experience solving previous similar problems. Common sense disputes are the kinds people run into from day to day. Children quarrelling over possession of objects, colleagues needing the same resource at the same time, and disputes encountered in reading the newspapers are just a few of kinds of disputes the program deals with. The MEDIATOR program, developed by Bob Simpson, begins with a semantic memory detailing the kinds of disputes it might encounter (e.g., physical, economic, and political) and a set of common mediation plans (e.g., one cuts the other chooses, split the difference, divide by parts). As it resolves disputes, it builds up an episodic memory organized by the concepts in its semantic memory. During processing, it first attempts recourse to previous experience to resolve a problem, and if no applicable experience is available, it uses default means (based on exhaustive search of alternatives) to resolve the problem. It learns based on feedback about the decisions it has made. If feedback is positive, it reinforces its belief that a particular type of plan is appropriate to a particular problem by storing the case and the plan used to resolve it. When it encounters later problems with features similar to one it has stored in memory, it will be reminded of that case and check to see if the plan used there was appropriate to its new problem. A positive experience may thus provide a shortcut in later problem solving. If feedback is negative, the MEDIATOR tracks down its error, fixes the knowledge that was responsible and attempts resolution of the problem a second time based on the new knowledge learned during feedback and the corrected knowledge that caused the previous error. When it finally resolves the problem satisfactorily, and stores the entire case in memory, later reminding of that case will (1) allow the problem solver to resolve a later similar problem without making the same mistakes a second time or (2) help the problem solver to figure out what went wrong when a similar failure occurs in the future.

There are several novel aspects to the MEDIATOR project. First, its model of problem solving includes not only the planning part of problem solving, but also problem understanding and failure resolution based on feedback. Case-based reasoning facilitates reasoning during all of these phases of problem solving.

Second, the analogical transfer process is "demand driven", where demand is provided by the task the problem solver is carrying out. When the problem solver is trying to classify a problem, it is the problem classification of the previous case that it inves-

tigates for applicability to the new problem. When it is attempting to derive a skeletal plan, it is the abstract plan from the previous case that it checks for applicability.

Third, the MEDIATOR has a well-articulated long term memory for experience. Problem solving experiences presented to the MEDIATOR are indexed in memory by those features which differentiate them from other experiences represented in similar ways. The memory organization is based on MOPs [2], [5].

A fourth novel feature of the MEDIATOR is in its use of the same problem solving model to both solve domain problems (resolving disputes) and to track down and fix failures in reasoning. It is able to do this because it treats both types of problems as first, classification problems, and then, plan instantiation problems. In solving domain problems, it thus seeks to classify disputes it encounters according to whether they are physical, economic, or political disputes during the understanding phase of problem solving. Each of these dispute types "knows" which types of plans are commonly useful to its resolution. Thus, classification allows pointers to potentially applicable skeletal plans, which are then refined for the particular problem.

During failure resolution, the MEDIATOR treats the failure it has encountered as its new problem. During the understanding phase of failure resolution (explaining the failure), it attempts to classify the error (as, e.g., a classification error, an elaboration error, a particular kind of elaboration error, a plan refinement error). Each of those error classifications has remediation plans associated with it to fix the faulty knowledge or faulty reasoning rule. It fixes its errors by instantiating and refining a plan appropriate to the kind of error it encountered (e.g., one can fix elaboration errors by using an alternate inference rule or by asking the value of a feature from the user). In the same way previous experiences can provide shortcuts in problem solving, previous failures can provide shortcuts in error recovery (e.g., the orange dispute above). This method of failure recovery has potential in domains where the types of failures that may be encountered and ways of recovering from each can be specified.

OTHER PROJECTS

While the MEDIATOR explores the framework for integrating learning, problem solving, and analogy, there are details that it does not address. One of those is determining the level of abstraction at which an analogical transfer should be made. We are addressing that problem in the domain of trouble shooting (and fixing) breakdowns in household appliances. Another topic not addressed is control of the simultaneous processes of problem solving and memory traversal. In our newest project, we are attempting to develop an architecture for problem solvers which use and learn from experience. We are concentrating on the interactions (via a blackboard) between three processes: the memory traversal process, the problem solver, and the interrupter. Our emphasis right now is in determining what the interrupter needs to know in order to decide that it is appropriate to interrupt the problem solver and present it with a case (found by the memory traverser).

REFERENCES

[1] Kolodner, J. L., "Towards an Understanding of the Role of Experience in the Evolution from Novice to Expert," in International Journal of Man-Machine Systems, November, 1983.

[2] Kolodner, J. L., Retrieval and Organizational Strategies in Conceptual Memory. Lawrence Erlbaum Associates, Hillsdale, NJ, 1984.

[3] Kolodner, J. L., and Simpson, R. L., "Experience and problem solving: a framework," in Proceedings of the Sixth Annual Conference of the Cognitive Science Society, Boulder, CO., 2-9, 1984.

[4] Kolodner, J. L., Simpson, R. L., and Sycara-Cyranski, K., "A Process model of case based reasoning in Problem Solving," in Proceedings of IJCAI-85, 1985.

[5] Schank, R. C., Dynamic Memory: A Theory of Learning in People and Computers, Cambridge University Press, London, 1982.

[6] Simpson, R.L., " A Computer Model of Case-Based Reasoning in Problem Solving: An Investigation in the Domain of Dispute Mediation," Ph.D. Thesis, Technical Report #GIT-ICS-85/18, School of ICS, GA Institute of Technology, Atlanta, GA, 1985.

ACKNOWLEDGEMENTS

This research has been supported in part by NSF Grant No. IST-8317711, in part by the Air Force Institute of Technology, and in part by ARO Grant No. DAAG29-85-K-0023. Robert Simpsons's current address is ARPA/IPTO, 1400 Wilson Blvd., Arlington, VA 22209.

G-26-617

The Role of Experience in Common-Sense and Expert Problem Solving

Final Report

Janet L. Kolodner, PI
School of Information and Computer Science
Georgia Institute of Technology
Atlanta, GA 30332

ARO Proposal Number: 21697-MA
ARO Contract Number: DAAG29-85-K-0023
Period of Contract: December 1, 1984 - Nov. 30, 1987

1 Problem Statement

The objective of this research was to elucidate the role of experience in common-sense and expert problem solving. Our aim was to discover and describe the processes involved in extracting useful conceptual knowledge from experience, in organizing and building the schemata to hold that knowledge, and in using that information in problem solving. In research areas as diverse as natural language processing and expert systems, researchers are plagued by the fact that the knowledge the systems need is hard to collect and input to the system. One way this bottleneck, called knowledge acquisition, can be relieved is by providing systems with a means of learning from their experiences. This research helps to lay the theoretical foundation for reasoning systems that (1) can become more expert through experience, (2) can make predictions and give advice based on previous experience in similar situations, and (3) can adapt to changes in their environments.

2 Background

In the work done under this contract, we have focussed on a problem solving technique called *case-based reasoning* (Hammond, 1986, Kolodner & Riesbeck, 1986, Kolodner & Simpson, 1984, 1985, 1988, Kolodner, et al., 1985, Kolodner, 1983, 1985, 1987a, 1987b, Rissland, 1982, Simpson, 1985). In case-based reasoning, a problem solver remembers previous similar situations and uses what it remembers about those situations to solve its new problem. Noticing similarities between experiences allows a problem solver to solve problems more efficiently, while remembering similar situations that resulted in failure allows a problem solver to anticipate and avoid failures in solving a new problem.

Our investigation has been primarily in the task domain of mediation, a complex real-world domain. We have consulted with experts to find out how they solve problems in this domain, and we have constructed a serires of progressively more sophisticated computer programs that model some of the processes involved in mediation. Our programs, called the MEDIATOR and

the PERSUADER, take as input the demands made by both sides in various disputes. Based on knowledge of previous contractual agreements, the programs each classify the current dispute with respect to other disputes with which they are familiar, and suggest solutions. Simulated feedback from both parties forces each program to repair its initial suggestion to be more in line with the previously unknown demands of the disputing parties. Each program then remembers its experience so that in later cases it can take shortcuts in problem solving and avoid previously-made mistakes.

The MEDIATOR (Kolodner & Simpson, 1984, 1985, 1988, Kolodner et al., 1985, Simpson, 1985) solves resource disputes concerning one disputed object in a common-sense way. The PERSUADER (Sycara, 1985a, 1985b, 1987a, 1987b, 1987c) solves labor mediation disputes similarly to the way a human mediator does. While the MEDIATOR showed the usefulness of case-based reasoning for a complex task and showed several functions a case-based reasoner can perform, the PERSUADER shows how case-based reasoning can be used for partial satisfaction of several competing and conflicting goals, and shows an instance of case-based reasoning being combined with analytic methods.

An example from the MEDIATOR's domain will illustrate case-based reasoning. We assume our hypothetical reasoner starts with the "book knowledge" we expect a novice to have. Through experience, that book knowledge becomes more refined, its domain of applicability is learned, and the previously unrelated facts become related and therefore more useful. The example is in the domain of mediation of common-sense disputes. A failed mediation attempt triggers a need to explain the failure. A later episode, in a different domain, but with the same goal structure, causes reminding of the first episode, and through case-based reasoning, a prediction and advice about a proposed solution are given.

> Two sisters are quarrelling over an orange. Their mother surveys the situation and proposes that each sister take half of the orange. One of the sisters complains, since she wants to use the whole peel for baking. Realizing the real nature of the conflict, the mother suggests that the sisters divide the orange agreeably: one will take the fruit and eat it, while the other will take the peel and use it for baking.

Analysis of this example shows that while the mother thought that both sisters had the same goal, she was mistaken. Though their sub-goals were in conflict, their goals were not. Stepping back and considering the real goals rather than the manifest ones resulted in a goal concordance. The following shows how this analysis is transferred in understanding and making a prediction about another situation. Here, we imagine the mother reading the following story in the paper:

> Egypt and Israel both want possession of the Sinai. The US suggests they cut it down the middle. Both Egypt and Israel complain.

Analogy to the orange dispute allows her to conclude that possession of the Sinai is merely a subgoal, that the real goals of the two countries should be considered, and that a mutually-agreeable split based on those goals be sought. This interpretation of the Sinai Dispute is done by *case-based reasoning.*

3 Results

Case-based problem solving uses previous experiences to suggest means of solving new problems. Recall of a previous experience can aid in understanding the intracasies and focus of a new problem, generating a plan for resolution of the problem, and in case of failure, in explaining and remedying the failure and re-evaluating the case. Recall and application of knowledge gained in dealing with previous novel cases can cut down the amount of reasoning necessary to resolve a new problem and can prevent failures from being repeated. In essense, case-based reasoning involves recall of a previous case, focus on those parts of the previous case that can be helpful in solving the new problem, and analogical transfer and then modification of some portion of the previous case to solve the new problem.

Case-based inference, in the simplest case, requires the following steps (Kolodner, 1987b, Kolodner & Simpson, 1988):

1. Recall a previous case.
2. Focus on appropriate parts of the case.
3. Adapt the focused-on parts of the previous case to fit the new case.

Recall of a case is done by probing the case memory. This is usually done several times during problem solving. Our programs probe memory each time they have a new goal to achieve. As the problem to be solved gets better defined, more specific cases become available. Thus, several cases may be used in the course of solving a single problem. In general, memory returns several cases rather than just one. Thus, the recall step also involves a filtering step in which the best-matching case of those retrieved from memory is selected.

Because any case that is recalled can be quite large, a case-based reasoner must be able to focus on the parts of the previous case that will be helpful in solving the new problem. This can be done by using the goals of the problem solver with respect to the new case. In short, focus is directed at those parts of the previous case that achieved the goal analogous to the one that must be achieved for the new case.

Because no two cases match exactly, the solution to a previous case is not usually exactly applicable to the new case. Thus, a case-based reasoner is responsible for adapting the parts of the previous case to fit the new case. In the simplest problems, there is no adaptation, and this step is merely a transfer step. In some situations, the method by which the old solution was derived is transferred to the new case, in some situations, domain-specific adaptation heuristics are applied, and in some situations, domain-independent adaptation heuristics are used. The case-based reasoner in effect acts as a *hypothesis generator* during the focus step, proposing possible ways to achieve the problem solver's goals, and acts as *hypothesis adapter* in the third step, turning the coarse proposals made by a previous case into solutions applicable to the new problem.

3.1 The Case-Based Reasoning Paradigm: The MEDIATOR

The major contribution made by the MEDIATOR project was in defining the problem solving paradigm underlying case-based reasoning. The MEDIATOR (Kolodner & Simpson, 1984, 1985, 1988, Kolodner, et al., 1985, Simpson, 1985) resolves common sense disputes based on experience solving previous similar problems. By common-sense disputes, we refer to the kinds people run into from day to day. Children quarrelling over possession of objects, colleagues needing the same resource at the same time, and disputes encountered in reading the newspapers are just a few of kinds of disputes the program deals with. The MEDIATOR program, developed by Bob Simpson, begins with a semantic memory detailing the kinds of disputes it might encounter (e.g., physical, economic, and political) and a set of common mediation plans (e.g., one cuts the other chooses, split the difference, divide by parts). As it resolves disputes, it builds up an episodic memory organized by the concepts in its semantic memory. During processing, it first attempts recourse to previous experience to resolve a problem, and if no applicable experience is available, it uses default means (based on exhaustive search of alternatives) to resolve the problem. It learns based on feedback about the decisions it has made. If feedback is positive, it reinforces its belief that a particular type of plan is appropriate to a particular problem by storing the case and the plan used to resolve it. When it encounters later problems with features similar to one it has stored in memory, it will be reminded of that case and check to see if the plan used there was appropriate to its new problem. A positive experience may thus provide a shortcut in later problem solving. If feedback is negative, the MEDIATOR tracks down its error, fixes the knowledge that was responsible and attempts resolution of the problem a second time based on the new knowledge learned during feedback and the corrected knowledge that caused the previous error. When it finally resolves the problem satisfactorily, and stores the entire case in memory, later reminding of that case will (1) allow the problem solver to resolve a later similar problem without making the same mistakes a second time or (2) help the problem solver to figure out what went wrong when a similar failure occurs in the future.

There are several novel aspects to the MEDIATOR project. First, its model of problem solving includes not only the planning part of problem solving, but also problem understanding and follow-up based on feedback. Problem understanding must be included as part of problem solving because problems specifications are often incomplete and ambiguous. Follow-up procedures are necessary in order for learning to happen. If a problem was solved successfully, follow-up might only include indexing the case appropriately in memory so that it can be recalled in future similar circumstances. If some error occured as a result of problem solving, follow-up procedures include explaining the reason for the failure and recovering from it or figuring out how it could have been avoided. It is these follow-up procedures that allow a problem solver to learn from its experience.

Second, the MEDIATOR was the first implemented case-based reasoner and showed several uses of case-based reasoning during problem solving. As illustrated in the MEDIATOR, case-based reasoning can facilitate reasoning during any of the problem solving tasks listed above. During problem understanding, previous cases can aid in classifying a problem and elaborating it. During plan generation, case-based reasoning is used to choose planning policies, to devise skeletal plans, to choose the actions, objects, and characters that take part in the plan, and to generate predictions about the results of executing a plan. During follow-up, previous cases can aid in assigning blame for an error and in choosing a method of recovering from a mistake.

Third, the MEDIATOR showed how the appropriate parts of a previous case can be focussed on during case-based reasoning. Focus in the MEDIATOR is "demand driven", where demand is provided by the goal the problem solver is attempting to achieve or the task it is attempting to carrying out. When the problem solver is trying to classify a problem, it is the problem classification of the previous case that is focussed on. When it is attempting to derive a skeletal plan, it is the abstract plan from the previous case that it checks for applicability. Since transfer of information from one case to another derives from this focus, the analogical transfer of information from one case to another can also be said to be driven by the demands of the problem solver.

Fourth, the MEDIATOR has a well-articulated long term memory for experience. Problem solving experiences presented to the MEDIATOR are indexed in memory by those features which differentiate it from other experiences stored there. The memory organization is based on MOPs (Kolodner, 1984, Kolodner & Cullingford, 1986, Schank, 1982).

A fifth novel feature of the MEDIATOR is in its use of the same problem solving model to both solve domain problems (in this case, to resolve disputes) and to track down and fix failures in reasoning. It is able to do this because it treats both types of problems as first, classification problems, and then, plan instantiation problems. In solving domain problems, it thus seeks to classify disputes it encounters according to whether they are physical, economic, or political disputes during the understanding phase of problem solving. Each of these dispute types "knows" which types of plans are commonly useful to its resolution. Thus, classification allows pointers to potentially applicable canned plans, which must then be refined for the particular problem.

During failure resolution, the MEDIATOR treats the failure it has encountered as its new problem. During the understanding phase of failure resolution (explaining the failure), it attempts to classify the error (as, e.g., a classification error, an elaboration error, a particular kind of elaboration error, a plan refinement error). Each of those error classifications has remediation plans associated with it to fix the faulty knowledge or faulty reasoning rule. It thus fixes its errors by instantiating and refining a plan appropriate to the kind of error it encountered (e.g., one can fix elaboration errors by using an alternate elaboration rule or by asking the value of a feature from the user). In the same way previous experiences can provide shortcuts in problem solving, previous failures can provide shortcuts in error recovery (e.g., the orange dispute above). This method of failure recovery has potential in domains where the types of failures that may be encountered the of known ways of recovering from each can be specified.

3.2 Precedent-Based Reasoning: The PERSUADER

While the MEDIATOR showed the usefulness of case-based reasoning and pointed out some important aspects of problem solving, the PERSUADER provided a more in-depth investigation of the processes involved in transferring information from one case to another. The PERSUADER's transfer method is a specialization of case-based reasoning called *precedent-based reasoning*. Prededent-based reasoning is a method of deriving a solution to a new case by recalling one that is highly similar, computing the differences between the recalled and the new case, and based on those differences modifying or patching the old solution to fit the new situation. Case-based reasoning can also be used for this last step. Precedent-based reasoning, as implemented in the PERSUADER (Sycara, 1985c, 1987a, 1987d, 1988b), involves the following steps:

1. Recall a previous similar case to act as "precedent".

2. Do a "coarse adaptation" or "adjustment" of the results of the previous case to create a "ballpark solution" to the new problem. The ballpark solution takes only a set of coarse-grained features into account, but does not deal with details. It is meant to compensate for the dissimilarity between the recalled precedent and the "ideal" precedent, if it existed.

3. Evaluate the ballpark solution to see if it can achieve (or partially achieve) the set of goals it is designed to achieve given the context of the current problem. Three categories of knowledge are used here: more detailed knowledge about the problem itself, knowledge of the problem solving context (i.e., the environment in which the problem is being solved) and its effects on the situation, and knowledge of past failures in similar situations.

4. Using a set of task-&-domain-specific heuristics coupled with previous experience, do a detailed modification of the ballpark solution to create a solution that will work in the current problem solving context.

The PERSUADER uses case-based reasoning to resolve labor management disputes. Mediation, in these situations, is an iterative process. The mediator first attempts to ascertain the goals of the disputants, then attempts to construct a reasonable solution to the dispute. Often, the presentation of the "reasonable" solution to the disputants elicits additional constraints from the disputants about the problem, and the mediator is forced to modify the solution or construct a new solution to fit the better-defined problem. This process might go on for several cycles. When the mediator is sure that it/he/she has a full understanding of the problem and has created the best possible solution, a process of argumentation is used to persuade one or both disputing parties to agree to a proposed solution (Sycara, 1985a, 1985b, 1985d).

The PERSUADER uses precedent-based case-based reasoning for a variety of tasks: to create an initial solution to a dispute, to resolve impasses brought about by a disputant who will not agree to a proposed solution, and to derive arguments of persuasion that are used in an attempt to persuade a recalcitrant party to agree to a solution. For each of these, the particular features of a case that differentiate it from an "ideal" precedent are different (step 2), the particular features used for evaluation are different (step 3), and the set of task-&-domain-specific heuristics are different (step 4), but the general process remains the same.

The PERSUADER shows in detail how precedent-based reasoning works for a particular domain (labor mediation), and just as importantly, shows under what circumstances it breaks down and what can be done when that happens. When no cases are available, the program employs analytic methods, in this case an adaptation of utility theory formulations that we call "preference analysis" (Sycara, 1987a, 1987b) to mediate between goals and come up with a compromise solution. Any program that uses case-based reasoning will need some kind of "from-scratch" method when cases are not available, and one appropriate to the particular domain must be chosen. When a case is so atypical that neither precedent-based reasoning nor a from-scratch method of dealing with normal cases from a domain (in this case, preference analysis) can be used, some way of using domain-independent knowledge must be used. The PERSUADER uses "situation assessment" (Sycara, 1987a, 1987c, 1987d), a method of case-based reasoning in which domain-independent knowledge describing an analogous causal situation is used. Each is explained briefly below.

Preference analysis is a process that takes the relative utility of the goals of each of the disputants in a dispute into account to measure the potential for agreement to a proposed contract. It is used to come up with a contract if no precedent is available, to evaluate several potential contracts with respect to each other, and to judge which tradeoffs might be appropriate when everybody's goals cannot be fulfilled. It is reported in detail in (Sycara, 1987a, 1987b).

Even when cases are available, precedent-based reasoning methods may not be appropriate. This is the case when the new case is different from what is expected in ways that predict that the usual types of solutions won't work. In labor/management disputes this happens when the company is being mismanaged, when the union or the company have goals that are out of line with the norms, and several other times. The PERSUADER's way of dealing with this type of situation is to classify it by its goal/plan interactions (much as Schank suggests in his formulation of TOPs), and to use knowledge about dealing with those abstract kinds of situations to solve the problem (Sycara, 1987a, 1987c). For example, if the company is being mismanaged, one applies "mismanagement remedies" in coming up with a solution. One mismanagement remedy is to punish those who are doing the mismanagement by placing an overseer over them to make sure they will do things correctly in the future. In the labor/management domain, this might translate into placing union members on the board of directors. An interesting aside to this method is that while it is hard in general to specialize general strategies or remedies to specific new kinds of situations, once it has been done the case can be remembered and case-based reasoning can also help here.

The methodology used in the PERSUADER integrates analytic methods (preference analysis) with heuristic methods (precedent-based reasoning and situation assessment) to create a highly robust problem solver (Sycara, 1987a). There are several ways the heuristic and analytic methods interact. The analytic method provides a way to construct a solution when heuristic methods cannot be used. The heuristic methods support the analytic by providing necessary information that would be tedious to obtain otherwise. The analytic method provides a means to evaluate a solution constructed by heuristic methods. The integration of analytic and heuristic methods provide the following advantages for the PERSUADER:

1. The problem solver does not break down when heuristic methods fail.
2. The problem solver can flexibly apply the most natural solution method to each problem it encounters, sometimes using a variety of methods to solve a single problem as the problem evolves.
3. Heuristic methods can be used to construct a ballpark solution, while analytic methods can be used to refine it to a detailed level if heuristic methods are incapable of doing that.

The PERSUADER's model, as a whole, presents models of (a) resolution of multiple conflicting goals, (b) planning for partial goal satisfaction, (c) persuasive argumentation, and (d) integration of heuristic and analytic methods. As a model of conflict resolution, the PERSUADER suggests what the ingredients of resolution strategies must be. As a model for partial goal satisfaction, it has implications for human decision making. As a system that embodies a theory of persuasive argumentation, it presents a novel framework for the study of attitude and belief modification. It also demonstrates the usefulness of case-based reasoning in a variety of tasks necessary for problem solvin gin complex domains. The novelty of the research is not only that it addresses problems little studied before, but also that it addresses them in an integrated framework.

3.3 Other Issues in Case-Based Reasoning

There are several additional case-based reasoning issues that we have addressed over both of these projects: what gets transferred during case-based reasoning and what types of case-based reasoning processes do that transfer, anticipating and avoiding previously-made mistakes, and representing cases.

3.4 Transfer and Adaptation Processes in Case-Based Reasoning

There are several processes that we have identified for making case-based inferences (Kolodner, 1987b):

1. Transfer the solution that achieved the current goal in the previous case.
2. Transfer the solution that achieved the goal and modify it based on differences between the current and previous cases.
3. Transfer the inference method by which the previous goal was achieved.
4. Create an abstraction of the problem descriptions of the old and new caes, extend it to fit the solution to the previous case, apply the abstraction to the new case to create the framework for a solution, and refine that framework to fit the new case.

The process to be used depends on a number of considerations. Process 1 is used when the goal to be achieved can be achieved by choosing a single value or fully-instantiated frame. This method is simplest, and is employed by the MEDIATOR. Process 2 is precedent-based reasoning, employed by the PERSUADER. It is appropriate when there are several goals to be achieved simultaneously, when the previous solution integrates the achievement of several goals simulateously, or when the problem solver's goal is not one that is easily decomposable into non-overlapping parts.

Process 3 (Kolodner, 1986, 1987b) is useful when the details of the old and new cases are so different that no particular features of the old case can be transferred to the new, but the environmental factors (e.g., constraints) that would be used to choose a plan to achieve the current goal are similar. In this case, the inference method used previously is used to achieve the goal in the new case. While neither the MEDIATOR nor the PERSUADER use this method, it is implemented in another program, called JULIA, that plans meals. JULIA uses this method if, for example, it is asked to plan a vegetarian meal. Upon remembering a previous meal where the main course was chosen by selecting a main course central to the specified cuisine and then finding a vegetarian recipe for it, JULIA is able to choose a main course for another vegetarian meal with a different cuisine by this method.

Process 4 (Kolodner, 1987b, Shinn, 1988a, 1988b) is an analogy method. In this method, a mapping is made between the problems of the current and previous cases. This mapping is used to create a solution schema that describes both cases. This solution schema will be an abstraction of the two cases. It is then applied to the new case, creating an abstract solution which must then be refined. This method subsumes the other methods, as it can transfer a solution directly, transfer

a solution method, or transfer an abstraction of a solution that is then modified. It is, however, a more time-consuming process, and one that we would want our automated reasoners to do only if the easier methods are not directly applicable.

3.5 Anticipating and Avoiding Mistakes

Previous failures serve several purposes during problem solving (Kolodner, 1987a, 1987b). They provide warnings of the potential for failure in the current case, and they may also provide suggestions of what to do instead. Analyzing the potential for failure in a new case, a necessary part of capitalizing on an old failure, may require the problem solver to gather additional information, thus causing the problem solver to change its focus of attention. A previous failed case that was finally solved correctly can help the problem solver to change its point of view in interpreting a situation if that is what is necessary to avoid potential failure.

Errors in reasoning can happen during any problem solving step. The problem might have been misunderstood initially, resulting in incorrect classification of the problem or incorrect inferences during the problem elaboration phase. Since problem understanding is an early part of the problem solving cycle, such misunderstandings and incorrect inferences propagate through to the planning phase, resulting in a poor plan. A problem might be understood correctly and all the necessary details known about it, but might still be solved incorrectly because poor decisions were made while planning a solution. In general, such errors are due to faulty problem solving knowledge. The problem solver might not have complete knowledge, for example, about under what circumstances a particular planning policy or plan step is appropriate. Finally, a problem might be solved correctly but carried out incorrectly by the agent carrying out the plan, or unexpected circumstances might cause execution to fail. Reminding of a case where any of these things happened warns the the problem solver of the potential for the same type of error in the new case. If the previous case was finally resolved correctly, details of its correct resolution are used to provide suggestions for solving the new problem correctly.

Any time the problem solver encounters a case with a previous problem, it considers whether there is the potential for that problem in the new case. This may cause it to refocus itself until the potential for failure is determined, and if such potential is determined and the problem solver has to retract decisions made previous to the current one, then it must remake any decisions dependent on those decisions. Such processing, of course, requires that the problem solver be integrated with a reason-maintenance system that keeps track of the dependencies among its decisions.

In short, the steps that must be followed to capitalize on a previous failure are (Kolodner, 1987a): (1) determine what was responsible for the failure, if possible (thismay already be recorded, and if not, some short amount of time is spent attempting toderive it), (2) direct reasoning focus tothe decision in the new problem that is analogous to the one that cause the failure in the previous one (this may be the one currently being focussed on or one that its correct solution is dependent on), (3) check for the potential for the same failure in the new case, either by seeing if the explanation of the previous failure holds in the new case or by checking the reasons why the previous decision was made and seeing if the same justifications might apply in the new case (this step may require additional information gathering), (4) if not, potential for error isn ot there, so return to the interrupted reasoning step and keep going, (5) if so, rule out the previous errorful decision as a

possibility for the current case, and if the previous case was finally resolved correctly, determine if the decision made when it was resolved correctly is applicable tothe new case, (6) if so, use it as a suggestion for a case-based inference, (7) if step 2 redirected reasoning focus, then redo whatever decisions must be redone as a result (i.e., by following dependencies) and return to the reasoning step that was interrupted.

3.6 Representing Cases

There are several representational issues that we have had to address to define our case-based reasoning processes appropriately. First we discuss the representational structure of cases. Then we discuss the knowledge that needs to reside with the solution part of a case.

Cases have five parts to them (Kolodner, 1985, 1987b, Kolodner & Simpson, 1988): (1) the problem being solved, stated in terms of goals to be achieved, constraints on those goals, and other environmental factors that go into choosing a solution, (2) the solution to the problem, including the reasoning that was done to come up with the solution and a set of predictions of what to expect if the solution is carried out correctly, (3) feedback from the world about what happened as a result of carrying out the solution proposed in (2), (4) evaluation of that feedback, and (5) next problem solving steps taken as a result of that evaluation (e.g., another case).

Because much of the processing in case-based reasoning requires knowing why previous decisions were made, what other decisions previous decisions were dependent on, and what was responsible for previous failures, there must be both a representational system and a bookkeeping system that keep track of that knowledge. In the systems we are building, we store this knowledge with the solution part of each case. In short, each value recorded in the solution has a *value frame* associated with it (Kolodner, 1986, 1987a). Each time the problem solver makes a decision, it records its decision in the value slot of the value frame and also records what led to the decision. This might include an inference rule that was applied and the set of values it was applied to. Value frames include facets for the chosen value, other values that were suggested as alternatives but not chosen, ruled out values, conditions that were taken into account in choosing a value, and the inference rule or set of steps used to make the decision. The knowledge kept in value frames supports both transfer of reasoning method from one case to another and avoidance of previously-made mistakes.

4 Conclusions

Our studies of case-based reasoning are showing that exploitation of previous experience provides considerable advantage to a problem solver. While there is much support structure needed for a case-based reasoner to do its work (a memory for cases, a reason maintenance system to keep track of dependencies, value frames to keep track of justifications and past reasoning), case-based reasoning allows a problem solver to exploit its experience to take shortcuts in reasoning and to anticipate and avoid previously-made errors. This might ultimately allow us to build expert and common-sense problem solving systems that can learn from both their successes and their mistakes. Of course, there are many problems we have not addressed here that must still be addressed: how

to best index cases so that the best ones are made available by the memory, which cases to keep in memory and how to organize them so the problem solver is not inundated with cases, how to choose the best of many cases provided by a memory, how to integrate the memory with the problem solver, how to integrate a case-based reasoner with other reasoners it needs to communicate with, processes for tracking down and explaining failures, processes for generalizing from both successful and failed problem solving experiences. While work is being done in each of these areas in research projects at Georgia Tech (see, e.g., Kolodner, 1983, 1985, Hinrichs, 1988, Turner, 1986, 1988, Shinn, 1988a, 1988b) and elsewhere (e.g., Alterman, 1986, Carbonell, 1983, 1986, Hammond, 1986, Kass, 1986, Rissland, 1982, Ross, 1982, Schank, 1982, Sycara, 1988a, 1988b), there is still considerable work to be done on all of these problems.

In 1984, when this project began, case-based reasoning was virtually unknown in the field of Artificial Intelligence. Partially as a result of the work done under this contract, case-based reasoning is becoming widely known within AI, and there is a great deal of interest in the use of case-based reasoning methods. In addition, Georgia Tech is now known as a leader in the area of case-based reasoning. While in 1984, our research group was composed of 4 students, we now have over a dozen students doing work related to case-based reasoning at Georgia Tech. This work is currently supported by the Army Research Institute for the Behavioral Sciences, the National Science Foundation, and Lockheed AI Center. Support from DARPA will begin in the next months.

5 References

1. Alterman, R. (1986). An adaptive planner. *Proceedings of AAAI-86.*

2. Carbonell, J. G. (1983). Learning by analogy: formulating and generalizing plans from experience. In Michalski, R. S., Carbonell,J.G. and Mitchell, T.M. (Eds.) *Machine Learning*, Tioga Publishing Co.

3. Carbonell, J. G. (1986). Derivational analogy in problem solving and knowledge acquisition. In Michalski, R. S., Carbonell, J. G., and Mitchell, T. M. (Eds.) *Machine Learning, Vol II*, Morgan-Kaufmann Publishers.

4. Hammond, K. J. (1986) *Case-Based Planning: An Integrated Theory of Planning, Learning, and Memory.* Ph.D. Thesis. Dept. of Computer Science, Yale University, New Haven, CT.

5. Kass, A. (1986). Modifying explanations to understand stories. *Proceedings of the Conference of the Cognitive Science Society.*

6. Kolodner, J. L. (1983). Towards an Understanding of the Role of Experience in the Evolution from Novice to Expert. *International Journal of Man-Machine Systems*, November, 1983.

7. Kolodner, J. L. (1984). *Retrieval and Organizational Strategies in Conceptual Memory.* Lawrence Erlbaum Associates, Hillsdale, NJ.

8. Kolodner, J. L. (1985). *Experiential Processes in Natural Problem Solving.* Report No. GIT-ICS-85/23. School of Information and Computer Science. Georgia Institute of Technology. Atlanta, GA.

9. Kolodner, J. L. (1986). Some little-known complexities of case-based inference. *Proceedings of the Third Annual Workshop on Theoretical Issues in Conceptual Information Processing*, April, 1986.

10. Kolodner, J. L. (1987a). Capitalizing on failure through case-based inference. *Proceedings of the Conference of the Cognitive Science Society*, 1987.

11. Kolodner, J. L. (1987b). Extending problem solver performance through case-based inference. *Proceedings of the Fourth International Machine Learning Workshop*, June, 1987.

12. Kolodner, J. L. and Cullingford, R. E. (1986). Towards a memory architecture that supports reminding. *Proceedings of the Conference of the Cognitive Science Society.*

13. Kolodner, J. L. and Riesbeck, C. (1986). *Experience, Memory, and Reasoning.* Lawrence Erlbaum Associates, Inc. Publishers, Hillsdale, NJ.

14. Kolodner, J. L., and Simpson, R. L. (1984). Experience and problem solving: a framework. *Proceedings of the Sixth Annual Conference of the Cognitive Science Society*, Boulder, CO., 2-9.

15. Kolodner, J. L. and Simpson, R. L. (1985). Using experience as a guide for problem solving. it Machine Learning Research. Kluwer Academic Press.

16. Kolodner, J. L. and Simpson, R. L. (1988). *The MEDIATOR: A Case Study of a Case-Based Problem Solver.* Report No. GIT-ICS-88/11 School of Information and Computer Science. Georgia Institute of Technology. Atlanta, GA, 1988.

17. Kolodner, J. L., Simpson, R. L., and Sycara-Cyranski, K. (1985). A Process model of case-based reasoning in Problem Solving. *Proceedings of IJCAI-85.*

18. Rissland, E. (1982). Examples in the legal domain: Hypotheticals in contract law. *Proceedings of the Fourth Annual Conference of the Cognitive Science Society.*

19. Ross, B. (1982). *Remindings and Their Effects on Learning a Cognitive Skill/* Cognitive and Instructional Sciences Series CIS-19. Palo Alto, CA: Xerox Palo Alto Research Centers.

20. Schank, R. C. (1982). *Dynamic Memory: A Theory of Learning in People and Computer.* Cambridge University Press, London.

21. Shinn, H. (1988a). Abstractional Analogy. Submitted to AAAI-88.

22. Shinn, H. (1988b). Analogical mapping for case-based reasoning. Submitted to the 1988 Conference of the Cognitive Science Society.

23. Simpson, R.L. (1985). *A Computer Model of Case-Based Reasoning in Problem Solving: An Investigation in the Domain of Dispute Mediation.* Ph.D. Thesis. Technical Report No. GIT-ICS-85/18. School of ICS, GA Institute of Technology, Atlanta, GA.

24. Sycara, E. (1985a). Persuasive argumentation in resolution of collective bargaining impasses. *Proceedings of the Seventh Annual Conference of the Cognitive Science Society*, August, 1985.

25. Sycara, E. (1985b). Arguments of persuasion in labor mediation. *Proceedings of IJCAI-85*, August, 1985.

26. Sycara, E. (1985c). *Precedent-Based Reasoning in Expert Labor Mediation.* Report No. GIT-ICS-85/22. School of Information and Computer Science. Georgia Institute of Technology. Atlanta, GA.

27. Sycara, E. (1985d). *Arguments of Persuasion: Two Papers.* Report No. GIT-ICS-85/19. School of Information and Computer Science. Georgia Institute of Technology. Atlanta, GA.

28. Sycara, E. (1987a). *Resolving Adversarial Conflicts: An Approach Integrating Case-Based and Analytic Methods.* Ph.D. Thesis. Report No. GIT-ICS-87/26. School of Information and Computer Science. Georgia Institute of Technology. Atlanta, GA.

29. Sycara, E. (1987b). Utility theory in conflict resolution. *Annals of Operations Research: Approaches to Intelligent Decision Support*, 1987.

30. Sycara, E. (1987c). Finding creative solutions in adversarial impasses. *Proceedings of the Conference of the Cognitive Science Society.*

31. Sycara, E. (1987d). Planning for Negotiation: A Case-Based Approach. *Proceedings of the DARPA Knowledge-Based Planning Workshop*, Austin, TX, December 1987.

32. Sycara, E. (1988a). Resolving Goal Conflicts via Negotiation. Submitted to AAAI88.

33. Sycara,E. (1988b). Using Case-Based Reasoning for Plan Adaptation and Repair. *Proceedings of the DARPA Workshop on Case-Based Reasoning.* May, 1988.

6 Publications and Technical Reports

1. Kolodner, J. L. (1985). *Experiential Processes in Natural Problem Solving.* Report No. GIT-ICS-85/23. School of Information and Computer Science. Georgia Institute of Technology. Atlanta, GA.

2. Sycara, E. (1985). Persuasive argumentation in resolution of collective bargaining impasses. *Proceedings of the Seventh Annual Conference of the Cognitive Science Society,* August, 1985.

3. Sycara, E. (1985). Arguments of persuasion in labor mediation. *Proceedings of IJCAI-85,* August, 1985.

4. Sycara, E. (1985). *Precedent-Based Reasoning in Expert Labor Mediation.* Report No. GIT-ICS-85/22. School of Information and Computer Science. Georgia Institute of Technology. Atlanta, GA.

5. Simpson, R. L. (1985). *A Computer Model of Case-Based Reasoning in Problem Solving: An Investigation in the Domain of Dispute Mediation.* Ph.D. Thesis. Report No. GIT-ICS-85/18. School of Information and Computer Science. Georgia Institute of Technology. Atlanta, GA.

6. Sycara, E. (1985). *Arguments of Persuasion: Two Papers.* Report No. GIT-ICS-85/19. School of Information and Computer Science. Georgia Institute of Technology. Atlanta, GA.

7. Kolodner, J. L. and Simpson, R. L. (1985). Using experience as a guide for problem solving. *Machine Learning Research.* Kluwer Academic Press.

8. Kolodner, J. L. and Riesbeck, C. (1986). *Experience, Memory, and Reasoning.* Lawrence Erlbaum Associates, Inc. Publishers, Hillsdale, NJ.

9. Kolodner, J. L. and Simpson, R. L. (1986). Case-Based problem solving: A framework. Experience, Memory, and Reasoning.

10. Kolodner, J. L. (1986). Some little-known complexities of case-based inference. *Proceedings of the Third Annual Workshop on Theoretical Issues in Conceptual Information Processing,* April, 1986.

11. Sycara, E. (1987). *Resolving Adversarial Conflicts: An Approach Integrating Case-Based and Analytic Methods.* Ph.D. Thesis. Report No. GIT-ICS-87/26. School of Information and Computer Science. Georgia Institute of Technology. Atlanta, GA.

12. Sycara, E. (1987). Utility theory in conflict resolution. *Annals of Operations Research: Approaches to Intelligent Decision Support,* 1987.

13. Sycara, E. (1987). Finding creative solutions in adversarial impasses. *Proceedings of the Conference of the Cognitive Science Society.*

14. Kolodner, J. L. (1987). Capitalizing on failure through case-based inference. *Proceedings of the Conference of the Cognitive Science Society,* 1987.

15. Kolodner, J. L. Extending problem solver performance through case-based inference. *Proceedings of the Fourth International Machine Learning Workshop*, June, 1987.

16. Sycara, E. (1987). Planning for Negotiation: A Case-Based Approach. *Proceedings of the DARPA Knowledge-Based Planning Workshop*, Austin, TX, December 1987.

17. Kolodner, J. L. and Simpson, R. L. (1988). *The MEDIATOR: A Case Study of a Case-Based Problem Solver.* Report No. GIT-ICS-88/11 School of Information and Computer Science. Georgia Institute of Technology. Atlanta, GA, 1988.

18. Sycara, E. (1988). Computer Aiding in Negotiation (Extended Abstract). Submitted to 2nd Conference on Computer Supported Cooperative Work.

19. Sycara, E. (1988). Multi-Agent Compromise via Negotiation (Extended Abstract). Submitted to the 8th Workshop on Distributed AI.

20. Sycara, E. (1988). Resolving Goal Conflicts via Negotiation. Submitted to AAAI88.

21. Sycara,E. (1988). Using Case-Based Reasoning for Plan Adaptation and Repair. *Proceedings of the DARPA Workshop on Case-Based Reasoning.* May, 1988.

7 Scientific Personnel and Advanced Degrees Awarded

1. Janet L. Kolodner, PI

2. Robert L. Simpson, Ph.D. student – not directly supported by the contract; however, computer time for his programming efforts (he programmed the MEDIATOR) and time spent by the PI advising his research were charged to this contract. Simpson received a Ph.D. in June, 1985 based on his work developing and describing the MEDIATOR and the problem solving paradigm it illustrated. He is now a Program Director at DARPA. Ph.D. Thesis: *A Computer Model of Case-Based Reasoning in Problem Solving: An Investigation in the Domain of Dispute Mediation.*

3. Ekaterini (Katia) Sycara, Ph.D. student. Received her Ph.D. in June, 1987 based on her work on the PERSUADER. Her graduate work was almost completely funded by this project. She is now a Research Associate in the Carnegie-Mellon University Robotics Institute. Ph.D. Thesis: *Resolving Adversarial Conflicts: An Approach Integrating Case-Based and Analytic Methods.*

4. T. Rangarajan, Ph.D. student – supported for two quarters in 1985. Was later dismissed from the project for lack of progress.

5. Thomas Hinrichs, MS student, Ph.D. student – supported for three quarters in 1986 and 1987. Worked on representing cases. Received his MS in 1987, partially funded by this project. He is now a Ph.D. student working on the use of case-based and other problem solving techniques to solve problems in open-worlds, Ph.D. expected in August, 1989.