Mixed-Initiative Control of Intelligent Systems

G.C. Borchardt
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA 91109

1. Abstract

Mixed-initiative user interfaces provide a means by which a human operator and an intelligent system may collectively share the task of deciding what to do next. Such interfaces are important to the effective utilization of real-time expert systems as assistants in the execution of critical tasks. This paper presents the Incremental Inference algorithm, a symbolic reasoning mechanism based on propositional logic and suited to the construction of mixed-initiative interfaces. The algorithm is similar in some respects to the Truth Maintenance System, but replaces the notion of "justifications" with a notion of "recency," allowing newer values to override older values yet permitting various interested parties to "refresh" these values as they become older and thus more vulnerable to change. The paper concludes with a simple example of the use of the Incremental Inference algorithm plus an overview of the integration of this mechanism within the SPECTRUM expert system for geological interpretation of imaging spectrometer data.

2. Introduction

Existing expert systems, in general, permit only a limited amount of initiative on the part of their human user. Typically, the user may provide hypotheses in the initial portions of a session, and following this the user will assume a strict role of question-answerer, providing evidence for or against various conclusions the system is attempting to reach, but always shielded from the higher-level decisions of the expert system. In such cases, the user is inherently the subordinate participant in the activity. On the other hand, there exist a great variety of software packages for such tasks as statistical analysis in which the software is completely subordinate to the user; that is, the user must provide all of the initiative.

In an ideal environment, one might envision a hybrid approach in which a human operator and an expert system or reasoning component of a software package deal more or less on an equal level, partitioning ongoing duties in whatever manner suits them, making independent decisions and keeping each other informed wherever their decisions overlap some area of particular concern to the other. The agents need not take explicit "turns," but could proceed at different rates, each providing input to a central coordination module. The division of labor agreed upon by the agents might even change over time or might involve mutual attention to given decision areas at times. Such an interface would be quite flexible, allowing the involvement of the attending expert system to range from a minimal level of interaction to total control of ongoing activity, depending upon the desires of the human operator. In this light, one sees mixed-initiative systems as filling a gap between the present status of expert systems and that of existing software packages for activities such as image processing, high-level robot manipulator control, and so forth.

In the area of Space Telerobotics, the mixed-initiative user interface approach seems especially useful due to the critical real-time nature of the activity coupled with the complications of detailed human guidance. In a mixed-initiative system for this activity, a human operator might retain control of those aspects which are of greatest importance to him, while relegating control of other aspects to an attending expert system where he lacks expertise, is too busy to provide concentrated attention, or has no access to relevant environmental factors. As a situation develops, the partitioning of duties might be adjusted to best utilize the particular capabilities and requirements of each of the agents.

The Incremental Inference algorithm is a symbolic reasoning mechanism geared especially toward use in the construction of mixed-initiative interfaces. The algorithm is based on the propositional calculus (that is, the formal language concerning the relation of various atomic "propositions" using connectives such as "NOT," "AND" and "OR") and allows any number of human or expert system "agents" to collectively share the role of overseeing the execution of a particular set of activities to be carried out by an external mechanism (usually an associated computer program). The algorithm is related to that of a Truth Maintenance System (see, for example, Doyle [1], McDermott and Doyle [2] and McAllester [3][4]) but differs in several key matters, particularly in the use of a "time value" associated with each proposition in place of the "justifications" associated with propositions in the Truth Maintenance System.

General use of the Incremental Inference mechanism in a mixed-initiative interface is as follows. Several agents communicate their intentions for activities or portions of activities to be performed by the system through providing "requests" to a centralized symbolic reasoning system, which computes the effects of such requests on the current understanding between the agents. The reasoning system gives priority to the effects of more recent requests over those of less recent requests and thus "overlays" previous conclusions with more recent ones, tracing the stated intentions of the agents as they progress in time. Each agent is also permitted to specify particular areas of "interest" within the reasoning system. The areas of interest to an agent may be changed as a session progresses and even inferred by the reasoning system itself. Given a particular set of current interests for each of the agents involved, whenever the logical consequences of any agent's requests overlap an area of interest to one of the agents, the interested agent is given the option of permitting the change as specified, or, alternatively, reasserting the status quo for the matter in question, thereby blocking the impending change.

The success of such an organization depends on two assumptions: (1) that in a mixed-initiative environment it is the desire of all agents to cooperate to a reasonable extent, and (2) that it is further the desire of all agents to partition the duties in an efficient manner, with at least one agent responsible for each area of decision-making. Given a division of labor such that the areas of concern for the agents are fairly disjoint, each agent may operate in a detached manner, not explicitly aware of what the other agents are doing. In other cases, such a division of labor is not possible or simply not desirable. In situations where the agents provide requests which overlap areas of concern to other agents, there is a high degree of intercommunication provided by the mixed-initiative interface constantly asking one agent, then another, if particular aspects of requested activities are to be accepted. The general framework in which the Incremental Inference system operates will become more apparent as the details of the algorithm are described.

3. The Incremental Inference Algorithm

In the design of a mixed-initiative user interface using the Incremental Inference mechanism, the initial step is to define a set of relevant "propositions," which may be thought of as state variables for constraining the range of possible activities. In a robotics system, for example, the propositions for the mixed-initiative interface might describe (1) high-level activities of the system such as "transport_object" and "couple_objects," (2) object type specifications such as "using_a_bolt," (3) individual objects such as "bolt_5" and "bracket_13," (4) high-level goals such as "repair_assembly_13," (5) plans for achieving those goals such as "replace_defective_components" and (6) various status conditions such as "phase_1_completed" and "bolt_removed."

Each proposition is associated with two values, a truth value, which may be "true," "false" or "retracted," and a time value, which is a number. Initially, all propositions are reset to a suitable state, such as "false" or "retracted" at time "0." As the session progresses, the propositions take on new truth and time values, so that at any time, the truth values of the propositions collectively describe the current state of the system, including the current activity in process. For example, using the above propositions, if "transport_object," "using_a_bolt" and "bolt_5" all have truth values of "true," then the system is currently in the process of transporting bolt 5. Time values are used to record a measure of recency for the truth values. Whenever a human operator or attending expert system asserts a new truth value for a proposition, the time value of that proposition is set to reflect the "current time," which may be "actual" time according to a particular clock or simply an integer value which is incremented with each submission of a new truth value for any proposition.

Logical relationships between individual propositions (e.g., the compatibility/incompatibility of various goals with various plans or various activities with various object types) are represented as propositional calculus formulas in conjunctive normal form. Conversion to conjunctive normal form produces a set of "clauses," which are disjunctions containing propositions and negations of propositions. For convenience within the Incremental Inference

system, the negation of a proposition is taken to be a proposition as well, with the constraint that complementary propositions must have compatible truth values (either both are "retracted" or one is "true" and the other "false") and must agree in time values. Thus, there is no distinction made between propositions and "literals" as is often the case in related symbolic reasoning systems.

Whenever the truth or time value of a proposition is changed, all clauses containing that proposition are reexamined, possibly resulting in changes made to other propositions. These changes occur as a result of two complementary processes, "inference" and "retraction." Inference occurs whenever there is a single proposition with oldest (least) time value within a clause, and none of the remaining propositions have a truth value of "true." In this case, the oldest-valued proposition is updated to the "true" status, with its time value updated to the next oldest time value found within the clause. Retraction occurs if several propositions, some of which are "false," share the oldest time value within a clause, and none of the propositions within the entire clause have a truth value of "true." In this case, all "false" propositions among those sharing the oldest time value are modified to have truth values of "retracted," with no change in their time values. (For a clause containing a single proposition, the inference and retraction processes behave as if a second proposition set to "false" at the current time is also contained within the clause.)

The processes of inference and retraction cover distinct cases: it is possible to perform inference or retraction or neither, but never both. When the truth and time values of a clause are such that neither inference nor retraction is required, the clause is said to be "stable." If inference or retraction is required, the clause is said to be "unstable." The process of examining a clause for which one of the contained propositions has been altered, possibly performing inference or retraction, is called "stabilization." The rule for stabilization of an arbitrary clause is summarized below.

STABILIZATION OF A CLAUSE "C":

(If "C" contains only one proposition, assume the existence of an additional proposition within "C," set to "false" at the current time.)

1.) (Possible inference.) If there is a single proposition "P" having the oldest time value within "C," and none of the remaining propositions are "true," update "P" to "true" at the next oldest time value found within "C."

2.) (Possible retraction.) If several propositions "P1," "P2," ..., "PN" (N > 1) share the oldest time value within "C" and there are no propositions with truth value "true" in "C," modify those of "P1," "P2," ..., "PN" which are "false" to "retracted," leaving their time values unchanged.

A reasonably efficient algorithm for the stabilization of a clause performs an initial scan through the clause, computing the oldest time value among its propositions, the number of propositions having this time value, the second oldest time value among the propositions and the newest time value for a "true" proposition within the clause. Following the determination of these quantities, it is a simple matter to decide which case applies and to perform the appropriate action.

As an example of the inference and retraction processes within the Incremental Inference mechanism, consider a single clause

(A v B v C),

with the following initial truth and time value assignments:

A = (retracted,0),
B = (retracted,0),
C = (retracted,0).

If "A" is updated to "false" at time "1," the clause remains "stable" (all propositions sharing the oldest time value are "retracted"), thus no inference or retraction occurs:

A = (false,1),
B = (retracted,0),
C = (retracted,0).

If following this, "B" is updated to "false" at time "2," the clause becomes unstable, resulting in the inference of a value of "true" at time "1" for "C":

A = (false,1),
B = (false,2),
C = (true,1).

Note that inference may occur even if one of the propositions in a clause is "true." If "A" is now updated to "false" at time "3", the following results:

A = (false,3),
B = (false,2),
C = (true,2).

If "C" is then updated to "false" at time "4," then "B" is updated to "true" at time "3," and so forth.

As an example of retraction, suppose that the same clause at some later point arrives at the following status:

A = (true,25),
B = (false,25),
C = (false,25).

If "A" is updated to "false" at time "30," then "B" and "C" are modified to the "retracted" status:

A = (false,30),
B = (retracted,25),
C = (retracted,25).

If "B" and "C" are contained in other clauses, their retraction might possibly propogate in the performance of other retractions. Additionally, the retraction of "B" and "C" might possibly cause other inferences within the system of propositions. It is a general feature of the Incremental Inference system that either inferences or retractions can, in the proper circumstances, lead to additional inferences or additional retractions. This aspect of the mechanism is discussed further in Section 5.

4. A Closer Look

Before proceeding, it is necessary to examine certain properties of the Incremental Inference mechanism in greater detail. One issue relates to the differences between the Incremental Inference system and the Truth Maintenance System. These differences are significant enough that they require some degree of clarification. It may be noted that in replacing the justifications of the Truth Maintenance System with time values as described above, it is not possible to perform actual "truth maintenance," retracting the inferred truth values of various propositions if later the support for these inferences subsides. Instead, there is a continuance of values inferred by the system as long as no future conclusions conflict with these values. A second difference arising from the use of time values in place of justifications is that it is thereby also not possible to perform "dependency-directed backtracking" as a means of resolving contradictions. The issue of backtracking is taken up in the following section in the context of "interests" coupled with the retraction process.

The continuance of inferred values within the Incremental Inference system may be thought of as subscribing to a version of the "STRIPS assumption," namely, that nothing is assumed to change unless some force explicitly compels it to do so [5]. This sort of assumption is quite practical in the context of mixed-initiative systems, where the object is to trace a sequence of states, as, for example, the state of performing a particular activity or pursuing a goal or implementing a plan. In dealing with a progression of this sort, it is reasonable to assume continuance in the absence of evidence to the contrary. Thus, states continue until new states supersede them. The corresponding

assumption is not practical, however, in the context of a Truth Maintenance System, where the object is the construction of logical proofs. The fact that an assertion was known to hold under a previous set of premises does not give us the power to infer that the same assertion holds under the current set of premises. This difference distinguishes the types of applications under which a Truth Maintenance System may be more appropriate from those under which an approach similar to that of the Incremental Inference system may be more appropriate.

Another issue requiring clarification involves the exact nature of the inference carried out by the Incremental Inference mechanism. In a similar fashion to that of McAllester's Reasoning Utility Package [4], inference performed by the Incremental Inference system is logically incomplete: that is, in some cases, the mechanism will fail to infer particular values for propositions when logically these values must follow from other values. In general, the incompleteness has to do with situations involving "case analysis" of alternative choices. In practice, the Incremental Inference system and related reasoning systems may be made to perform complete inference in particular setups through the inclusion of additional clauses within the system.

In all cases, the propagation of inference and retraction within the Incremental Inference system is guaranteed to dissipate within a finite amount of time. This is ensured due to the combination of a "ceiling" provided by the current time and a property of the stablization algorithm that whenever a proposition is modified in truth value, its time value is increased, the only exception being a change to the "retracted" status, for which the time value remains unchanged. Thus each proposition may carry at most two successive truth values ("true" or "false" plus "retracted") for each possible time value.

5. Additional Aspects of the Algorithm

One central aspect of the Incremental Inference mechanism has not been included in the above discussion. This aspect concerns the refreshing of time values for propositions and the specification of "interests" by the various agents participating in the overall decision-making process. Without the ability to "refresh" time values, or bring them up to the current time, there is an ever-increasing vulnerability of propositions to change as time progresses. The Incremental Inference mechanism is intended to be impartial, serving merely as a medium through which the various agents may coordinate their desired activities. Thus, the burden lies with the agents themselves for the protection of truth values for propositions of interest to them.

Accordingly, each agent is allowed to specify an "interest" in the collective propositions of any clause. Given an interest for a particular agent in a particular clause, if any proposition within that clause is to be modified in truth or time value, the agent is informed prior to the change and is given the opportunity to refresh the existing value, thereby bringing the time value of the proposition up to the current time. A few restrictions streamline this process: if the proposition is being updated in time value only or already has a time value equal to the current time, or if the proposition has "retracted" status prior to the change, the agent is not permitted to refresh the value and is merely informed of the change. The reasons for bypassing the permission-asking process in the first two cases are fairly obvious, as to attempt to block such actions by refreshing the time value to the current time makes little sense. The third case is more involved and concerns the nature of the "retracted" value. By permitting a proposition to assume the "retracted" status, an agent is considered to be approving either the value "true" or the value "false" for that proposition. A later query to the agent concerning the modification of that proposition from "retracted" to either "true" or "false" then appears to be redundant.

As a simple example involving interests and the refreshing of a proposition, consider the clause

(I v J v K v L v M),

with truth and time values at some point set to the following:

I = (true,45),
J = (false,45),
K = (false,48),
L = (false,50),
M = (false,59).

If agents "X" and "Y" have both expressed an interest in this clause and agent "X" requests the update of proposition "I" to "false" at a new current time of "60," the mechanism will proceed in the following manner.

1.) Agent "Y" will be asked for permission in changing "I" from "true" at time "45" to "false" at time "60." (Agent "X" is assumed to have given permission for the change in "I" by making the original request. Agent "X" will be asked for permission in all subsequent modifications resulting from this change, however.)

2.) If agent "Y" grants permission for the update in proposition "I," the mechanism will as a result attempt to infer a value of "true" at time "48" for proposition "J." Both agents "X" and "Y" will be asked for permission in this change.

3.) Suppose agent "X" permits the change in proposition "J" but agent "Y" does not. The truth value of "J" is then preserved as it stands, and the time value of "J" is incremented to the current time of "60." In a subsequent attempt to stabilize the clause, the mechanism attempts to infer a value of "true" at time "50" for proposition "K." Both agents are then asked for permission in the desired modification of "K."

4.) Suppose both of the agents grant permission. Proposition "K" is updated to "true" at time "50," and the clause is thus stablized.

At the conclusion of the above sequence of actions, the resulting status of the clause is as follows:

I = (false,60),
J = (false,60),
K = (true,50),
L = (false,50),
M = (false,59).

In large applications employing the Incremental Inference mechanism, it is perhaps most convenient to include a "premise controller" as described by McAllester in [3]. Such a device employs a number of set strategies (determined by each agent) in dismissing the majority of the permission requests without actually asking the agents; thus, only the most important or unusual requests actually pass along to the agents.

The specification of "interests" and the use of the refreshing device produce a number of different patterns of interaction for the agents. In some cases, an act of refreshing one or more propositions may directly conflict with the initially requested value, resulting in the subsequent retraction of the initial value itself. (This would be the case if agent "Y" had rejected the modification of all four remaining propositions, "J," "K," "L" and "M" in the above example.) The resulting retraction of the initially requested value may be taken as a sign of a direct incompatibility in the agents' desires. A reasonable action to take at this point is to restore the status of all propositions to their values as of a previous "stable point" for the entire system (e.g., before the last request or group of related requests). In other cases, the refreshing of particular values does not block a request completely; rather, the effect is that of narrowing down several choices to a few or in some cases a single remaining choice. As long as at least one such choice remains, however, the initially requested value will remain as well.

In the above, mention has been made of the interaction between inference and retraction: each may result in occurrences of the other. In the case of retraction, the effect is to generate all inferences and retractions which could be made if the proposition were either "true" or "false," subject to the particular configuration of clauses and the completeness limitations of the algorithm. That is, any proposition whose time value is older than that of a retracted proposition and whose truth value is not compatible with both a value of "true" and a value of "false" for the retracted proposition must be modified. In practice, this means that the retraction of a value will succeed in the complete propagation only if all agents are unconcerned with that value. In many cases, however, either the "true" aspect or the "false" aspect of the retracted value will be blocked through the refreshing of other values and thus the "retracted" value will later be modified to the remaining aspect, "false" or "true."

The process described above involving the retraction of values in combination with the refreshing of other values corresponds to the notion of "dependency-directed backtracking" in the Truth Maintenance System. Here, instead of computing the minimal set of assumptions underlying a "contradiction," or conflict of inferred values, a broader approach is taken in propogating all of the effects of keeping several options open simultaneously (these

options correspond to the retracted propositions). Through the use of the refreshing device, each agent may block certain options, possibly keeping others intentionally open. The combination of refreshing operations performed by the various agents determines which, if any, of several retracted propositions takes on a newly inferred value of "true" or "false."

Since the interests of the various agents may be expected to change during a session, the interest values are themselves implemented as propositions. Thus, it is possible for an agent to alter the specification of his current interests by modifying the truth values of the propositions which represent those interests. With interest values implemented as propositions, it is possible to construct systems utilizing several "layers" of the Incremental Inference mechanism. For example, a fundamental layer of propositions and clauses might describe the current state of activity for the task at hand, while an additional layer of propositions might describe various "interests" of the user regarding specific aspects of the ongoing activities. The propositions in the upper layer would also be constrained by clauses, thus making it possible to perform inferences about the user's interests. A separate layer above the fundamental layer might describe the interests of an attending expert system. Given this configuration, it is also possible to construct systems which reason about one agent's interests concerning another agent's interests, and so forth.

A less critical feature of the Incremental Inference mechanism, but nonetheless one which increases its overall efficiency, is that of providing a set of higher-level structures, called "decision sets," representing collections of clauses. By grouping sets of closely related clauses in this manner, it is possible in some cases to perform inference and retraction without considering the individual clauses separately. Such is the case with the four "decision set" varieties described below. Interest values may also be associated with these higher-level structures.

1.) "Unconstrained." Equivalent to a clause without the logical disjunction restriction. This type of decision set is used primarily as a means of grouping several propositions into a single set for the specification interests without affecting the logical structure of the system.

2.) "At least one." Equivalent to a single clause relating a set of propositions.

3.) "At most one." Equivalent to the N!/(N-2)!2! clauses containing the negations of pairs of propositions drawn from a common set. For example, a decision set of type "at most one" which contains the propositions "A," "B" and "C" is equivalent to the individual clauses "(NOT_A v NOT_B)," "(NOT_B v NOT_C)" and "(NOT_A v NOT_C)."

4.) "Exactly one." Equivalent to the combination of the clauses described by decision sets of the "at least one" variety and the "at most one" variety.

The above types cover all possible combinations of the presence and absence of constraints of the form "at least one" and "at most one." Inference and retraction for the constraint of the form "at least one" simply follow the stabilization rule given for clauses. A similar stabilization rule is provided below for the constraint of the form "at most one." This rule is equivalent in its effect to the corresponding stabilization of all clauses implied by a decision set of type "at most one."

STABILIZATION OF A DECISION SET "D" OF TYPE "AT MOST ONE":

1.) (Possible inference.) Modify all propositions with time values older than the newest "true" or "retracted" proposition(s) in "D" to "false" at the time of the newest "true" or "retracted" proposition(s).

2.) (Possible retraction.) If several propositions "P1," "P2," ..., "PN" (N > 1) share the status of being the newest "true" or "retracted" values in "D," modify those of "P1," "P2," ..., "PN" which are "true" to "retracted," leaving their time values unchanged.

A decision set of type "exactly one" may be stabilized by applying the rule for a decision set of type "at most one" followed by the rule for a decision set of type "at least one." (This ordering is necessary, as the "at most one" stabilization may produce a need for stabilization of the "at least one" type, but not vice-versa.) The decision set of type "exactly one" is extremely useful in compact representations of logical constraints. Similar constructions have been

used by Tenenberg in the area of resolution-based inference [6] and by Shapiro in his SNePS system [7].

6. A Simple Example

The following example involves the construction of a simple mixed-initiative interface using the Incremental Inference algorithm. The context is that of a simple robotics system in which there are three general modes of activity: "pausing," "transporting_an_object" and "rotating_an_object." Transporting an object may amount to either "carrying_an_object" or "sliding_an_object." Rotating an object may amount to either "twisting_an_object" (by a spinning of the wrist) or "inclining_an_object" (by a bending of the wrist). Finally, there are five objects, "bracket_1," "bolt_1," "bolt_2," "block_1" and "block_2," of which "block_1" is fixed in location and therefore may not be transported (it may, however, be rotated). All activities except "pausing" involve a single object; specification of an object is optional for "pausing."

Two agents, a human operator and a single attending expert system, coordinate the execution of activities in the simple robotics system. For simplicity, the interests of the two agents are assumed to be constant: the human operator is interested in the more general description of *what* is being done (transportation and rotation of objects), while the expert system is interested in the more specific description of *how* the activity is to be accomplished (carrying, sliding and so forth). Both agents are interested in the objects involved in the activities. The following is a set of function calls for setting up the constraints and associated interests as described above. (In the specifications appearing below, parentheses are used to enclose arguments to a function and brackets are used to enclose elements of a "list" data structure. The functions "unconstrained," "at_least_one," "at_most_one" and "exactly_one" initialize single decision sets of the specified types, taking a name and a list of elements for a decision set as arguments. The function "interest" specifies the interest of a particular agent in a particular decision set; the arguments are the decision set name, the agent and the interest value.)

```
exactly_one(ds1 [pausing transporting_an_object rotating_an_object])
interest(ds1 user true)

exactly_one(ds2 [not_transporting_an_object carrying_an_object sliding_an_object])

exactly_one(ds3 [not_rotating_an_object twisting_an_object inclining_an_object])

unconstrained(ds4 [carrying_an_object sliding_an_object twisting_an_object inclining_an_object])
interest(ds4 system true)

at_most_one(ds5 [bracket_1 bolt_1 bolt_2 block_1 block_2])
interest(ds5 user true)
interest(ds5 system true)

at_least_one(ds6 [pausing bracket_1 bolt_1 bolt_2 block_1 block_2])

at_least_one(ds7 [not_transporting_an_object not_block_1])
```

Suppose that the last activity performed at some point was that of carrying bolt 2. Thus, the propositions "transporting_an_object," "carrying_an_object" and "bolt_2" are "true" and the remaining propositions are "false." The following are three possible scenarios arising in a mixed-initiative user/system designation of the subsequent activity.

1.) The user requests that "rotating_an_object" be set to "true." The mechanism updates the current time and modifies the proposition accordingly. The user is then queried and gives permission for "transporting_an_object" to be modified from "true" to "false." Next, since "rotating_an_object" implies either "twisting_an_object" or "inclining_an_object," yet both are "false" at the same time and thus neither may be inferred as "true," an attempt is made to retract these values. The expert system, which is interested in both propositions, grants permission for the first of these retractions but rejects the second, resulting in a refreshing of the time value for "inclining_an_object," and, as a result of this, a further modification of "twisting_an_object" from "retracted" to "true."

Finally, the expert system is asked if "carrying_an_object" may be modified to "false," which it approves. The resultant state leaves "rotating_an_object," "twisting_an_object" and "bolt_2" "true" and the remaining propositions "false." Hence, the activity is to "twist" bolt 2.

2.) The user requests that "bolt_2" be set to "false." The mechanism has previously inferred from "transporting_an_object" that "block_1" must also be "false" but is unable to select a single new object from the remaining choices "bracket_1," "bolt_1" and "block_2" Thus, an attempt is made to retract these three propositions. The user rejects the retraction of "bracket_1" and the expert system rejects the retraction of "bolt_1." Both agents accept the retraction of "block_2," however, and since "block_2" is the only remaining possibility, it is not only modified to "retracted," but subsequently it is modified to "true." The resulting state leaves "transporting_an_object," "carrying_an_object" and "block_2" "true" and the remaining propositions "false." Hence, the activity is to carry block 2.

3.) The expert system requests that "bracket_1" and subsequently "inclining_an_object" be set to "true." In the ensuing propagation of values, the user grants permission for "bracket_1" to be modified to "true," "bolt_2" to "false," "rotating_an_object" to "true" and "transporting_an_object" to "false." The expert system grants permission for "bolt_2" and "carrying_an_object" to be modified to "false." The resulting states leaves "rotating_an_object," "inclining_an_object" and "bracket_1" "true" and the remaining propositions "false." Hence, the activity is to "incline" bracket 1.

In the above sequences, the user and the expert system each provide initiative in a number of ways. In the first sequence, the user specifies a general constraint for the next activity ("rotating_an_object"), and this constraint is made more specific by the expert system when it is queried concerning "twisting_an_object" and "inclining_an_object," resulting in an eventual activity of "twisting" bolt 2. In the second sequence, both the user and the expert system help narrow a field of three object choices to one for the subsequent carrying activity. Finally, in the third sequence, it is the expert system which provides the original initiative. In this sequence, it is noteworthy that the user gives permission for the changes not at the level of the propositions "inclining_an_object" and "carrying_an_object," but rather at the more general level of "transporting_an_object" and "rotating_an_object," which are of interest to the user.

In general, each cycle of interaction proceeds with one agent providing a request, this request either completely specifying or only partially specifying the subsequent activity. The remaining agents are queried concerning the modification of propositions of interest to them. Propositions for which permission is not given are refreshed to the current time, resulting in either a specialization of the originally requested activity or in some cases a complete blocking of the activity, resulting in a return of the mechanism to a previously saved state, most likely the point just before the last request. If the request is not completely blocked, other agents may provide additional requests. Eventually, all agents agree upon a final version of the activity specification and the activity is executed. The cycle then repeats with any of the agents providing a new initial request.

7. Use of the Mechanism in SPECTRUM

The Incremental Inference mechanism is currently being installed within the SPECTRUM expert system for geological interpretation of imaging spectrometer data. SPECTRUM has been in development at JPL for approximately two and a half years and is written using a combination of the programming languages STAR and C. The STAR programming language and its use in the implementation of SPECTRUM are described in [8] and [9]. SPECTRUM is intended to provide both expert assistance in the analysis of imaging spectrometer data and independent analysis of the data where appropriate.

Users of the SPECTRUM system may be expected to vary along several dimensions, including (1) background in Geology, (2) familiarity with imaging spectrometer data and its analysis, (3) familiarity with the site concerned by a given imaging spectrometer data set, and (4) familiarity with use of SPECTRUM. As a result, the Incremental Inference system is being used to create a mixed-initiative environment which may exploit particular areas of expertise in the user while providing assistance in areas complementing these areas of expertise. At the start of a session with SPECTRUM, the user will be asked to provide a rough specification of his general "interests" concerning the

four dimensions mentioned above. During the session, specific interests implied by the initial specification may be adjusted to a more comfortable level at any time by the user. At each juncture in processing the imaging spectrometer data, a mixed-initiative specification of the next activity takes place. In this manner, the user may take control of those aspects of the analysis which concern him the most, deferring control of other aspects to SPECTRUM. Decisions made by SPECTRUM must always pass through the Incremental Inference mechanism, however, so the user is ensured that, should particular aspects of the deferred matters be of interest to him, he will be asked for advice. Results of the integration of the Incremental Inference system into SPECTRUM will be discussed in a future paper.

8. Conclusion

Mixed-initiative user interfaces provide an important capability toward the construction of flexible expert systems which do not always insist on being experts. As well, mixed-initiative user interfaces may be applied in bringing passive software packages up to the level of knowledgeable assistants. In the long run, the distinction between expert systems and subordinate software packages may dissolve with the development of flexible computer programs capable of assuming a variety of roles ranging from passive subordinate to autonomous agent.

The Incremental Inference mechanism is apparently well-suited to the task of constructing mixed-initiative interfaces, due to its ability to trace a sequence of state changes while confirming various portions of the sequence with several interested parties. As well, application of the Incremental Inference algorithm may extehd beyond that of mixed-initiative systems to other areas of symbolic reasoning. For example, the mechanism might be used as a simple model of deliberation within a single agent or as an aid in simulation or plan execution monitoring.

9. Acknowledgements

The research described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration. The author would like to thank Dr. Jerry Solomon and Dr. Steven Vere for their comments regarding the Incremental Inference mechanism. Steven Groom implemented a large portion of the code. Additional members of the SPECTRUM design team include Miki Martin and Rachael Brady.

10. References

[1] Doyle, J., "A Truth Maintenance System," *Artificial Intelligence 12*, North-Holland, Amsterdam, 1979, 231-272.

[2] McDermott, D. and Doyle, J., "Non-Monotonic Logic I," *Artificial Intelligence 13*, North-Holland, Amsterdam, 1980, 41-72.

[3] McAllester, D. A., *An Outlook on Truth Maintenance*, MIT Artificial Intelligence Laboratory, Memo 551, Cambridge, MA, 1980.

[4] McAllester, D. A., *Reasoning Utility Package User's Manual, Version One*, MIT Artificial Intelligence Laboratory, Memo 667, Cambridge, MA, 1982.

[5] Fikes, R. E. and Nilsson, N. J., "STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving," *Artificial Intelligence 2*, North-Holland, Amsterdam, 1971, 189-208.

[6] Tenenberg, J. D., "Taxonomic Reasoning," *Proc. IJCAI-85*, Los Angeles, CA, 1985, 191-193.

[7] Shapiro, S. C., "The SNePS Semantic Network Processing System," in N. V. Findler (ed.), *Associative Networks*, Academic Press, New York, 1979, 179-203.

[8] Borchardt, G. C., "STAR: A Computer Language for Hybrid AI Applications," in J. S. Kowalik (ed.), *Coupling Symbolic and Numerical Computing in Expert Systems*, North-Holland, Amsterdam, 1986, 169-177.

[9] Borchardt, G. C., *STAR (Simple Tool for Automated Reasoning): Tutorial Guide and Reference Manual*, Jet Propulsion Laboratory, Publication 85-89, Pasadena, CA, 1985.