ELSEVIER Artificial Intelligence 76 (1995) 1-16 Artificial Intelligence

Planning: What it is, What it could be, An introduction to the Special Issue on Planning and Scheduling

Drew McDermott[a,*], James Hendler[b]
[a] Yale University, Department of Computer Science, Box 2158, Yale Station, New Haven, CT 06520, USA
[b] University of Maryland, Computer Science Department, A.V. Williams Building, College Park, MD 20742, USA

Planning is designing the behavior of some entity that acts, either an individual, a group, or an organization. The output is some kind of blueprint for behavior, which we call a *plan.* People make a lot of plans, sometimes for themselves, sometimes for other people, sometimes for machines. The question that defines the topic of this issue is: How can we automate planning? The question is interesting for the usual reasons. First, automating planning might shed light on how people and other animals design their behavior. Second, complex planning problems might be solved better with the aid of computers.

There are a wide variety of planning problems, differentiated by the types of their inputs and outputs. Typically, planning problems get more and more difficult as more flexible inputs are allowed and fewer constraints on the output are required. Typically, these problems get very difficult very quickly, as some of the papers in this issue will attest. The classical approach to planning problems is to start from specifications of the effects of actions, then try to infer a string of actions that bring about a particular state of affairs. In recent years many variations on this theme have been explored.

One important special case of planning is *scheduling,* for which the input to the behavior designer is a set of actions that must be carried out, and the output is an order in which to carry them out. Some orderings are better than others, because each of the given tasks will require a set of resources with a finite capacity. For example, in a transportation scheduling problem, we might be given a set of objects to move to various places, and a set of trucks to carry them in, and the desired output might be a schedule of movements that allows every truck to be as full as possible and idle as little as possible.

* Corresponding author. E-mail: mcdermott-drew@cs.yale.edu.

Elsevier Science B.V.
SSDI 0004-3702(95)00034-8

In this introduction, our main purpose is to survey the field of planning and scheduling, place our collection of papers in that context, and then to point out the field's weak and strong spots and some directions for the future. We'll start with a historical overview. Throughout this introduction, we use **this type face** when referring to papers in this special issue of *Artificial Intelligence.*

1. Development of the field: classical AI planning

When planning research started in the 1960s, it was mainly seen as an application of two standard AI techniques: search and theorem proving. In fact, at the outset there was little distinction made between search and planning, and confusion about the relation between the two persists to this day.

Search was seen as crucial to AI from the beginning, and is seen as crucial today. Many problems can be solved by applying a sequence of transformations starting from an initial null solution. At each step, there is usually a choice of which transformation to apply, most of which won't eventually lead to a complete solution, so it's necessary to keep track of partial solution states and return—or *backtrack*—to them when previous choices don't work out.

The "transformations" we have referred to are usually called *operators.* Our description of a search problem has been quite abstract; there is no commitment that operators correspond to actions an agent might take. For example, the problem of map coloring can be expressed as:

Initial state: A coloring that assigns no color to any country on the map.
Operators: Assign one more country a color that is not assigned to any adjacent country in the coloring so far.
Goal-state description: A coloring that colors every country.

However, if you think of operators as agent actions, then you get search problems such as this:

Initial state: An initial situation, e.g., "There are three blocks on a table, one red, one white, and one blue".
Operators: Actions a robot might take, e.g., "Move a block from the table to the top of another block".
Goal-state description: A desired situation, e.g., "There is a blue block on a white block and under a red block".

The very first planning system can be considered to be GPS, the General Problem Solver [16,37], because it could in principle solve any search problem, and some of the problems it solved looked like planning problems. An example (from [43]) is this: You are given a four-gallen jug and a three-gallon jug, and an unlimited supply of water. Find a sequence of fillings and pourings that get two gallons into the big jug.

Unfortunately, the possibility of expressing simple planning problems this way has led to the idea that this is the *only* way to think about planning problems, when in reality this approach to planning is applicable to only a small subset of these problems.

If we adopt this approach, then we:

- assume plans are sequences of actions,
- take the purpose of a plan to bring about a situation satisfying a description,
- treat the outcome of every action as perfectly predictable.

These assumptions essentially define what is now called *classical planning*. The assumptions have been questioned a lot recently, but actually they are quite reasonable in various applications. For example, planning a route through a city can be thought of as finding a series of blocks to traverse. It is not in fact perfectly predictable that the attempt to traverse a block will get you from one intersection to the next, but in most cases it is reasonable to treat it as predictable and worry about untraversable blocks when they are encountered.

The other strand that led through classical planning was the reduction of planning to theorem proving. In 1960, John McCarthy proposed the use of predicate-calculus reasoning to guide intelligent behavior, and the first big realization of this idea was Green's [23] program QA3, which solved a variety of simple problems expressed in predicate calculus. Among them was a set of planning problems, expressed in terms of McCarthy's *situation calculus* in which axioms about what actions led to what situations were used to deduce action sequences. These axioms embodied assumptions equivalent to those above. Unfortunately, just turning a theorem prover loose on the axioms led to a search problem that was hard to solve. Theorem provers look for chains of inferences that lead to conclusions, and these chains are only indirectly related to the chains of action we are interested in.

This problem with theorem proving was balanced by the advantage that expressing problems in terms of situation-calculus axioms was simple and clean. By contrast, GPS had the flaw that in expressing a new class of problems, it required the represention of not just the legal operators, but also of domain-dependent procedures for matching search states, and an "operator-difference table" that recorded which operators were relevant to reducing the differences found by the matcher. Creating all these procedures and tables was tedious, and often seemed to amount to giving the program too many hints.

In 1969, the AI group at Stanford Research Institute in Palo Alto, California, found a way to get the best of both approaches while avoiding many of the weaknesses. This group [19] devised a version of GPS that worked directly from action definitions stated in a form similar to that of the situation calculus. Each action was defined in terms of its *preconditions* and *effects*, stated as predicate-calculus atomic formulas. The action definition was used to *edit descriptions* of situation instead of *deducing properties* of situations. An action's effects were of two varieties, additions and deletions. Generating a new situation description from an old one was a matter of deleting all the atomic formulas in the delete list and adding all the ones in the add list. All other formulas in the old situation description were carried over.

This problem solver was called STRIPS ("STanford Research Institute Problem Solver"). It was able to solve bigger problems than previous approaches, and was used as the planner for the Shakey robot [18]. STRIPS remains influential, especially for the ideas it embodies about representation and temporal change.

STRIPS retains GPS's idea of keeping the search space close to situation space. Starting in the mid-seventies, the field of planning shifted away from that idea, initially for the purpose of improving search control, and later to allow for the solution of nonclassical planning problems. Although several programs tried variations on the STRIPS approach and/or on the details of how the space could be searched (see, for example the work of [46,51,54]), the first major break came with the programs NOAH [47] and Nonlin [55]. These programs explored a search space of *partial plans*, collections of plan steps that achieved some of the goals in the problem statement. Each plan step referred to a single action that would be part of the final plan. Actions have preconditions, which would become new *subgoals* to be achieved. Taking a step in the search space meant committing to achieving a subgoal with a new or existing plan step.

In these planners, the plan steps did not have to be kept in a linear order, and thus they have often been referred to as "nonlinear". More recently, it became clear that the term was somewhat misleading, and thus nowadays they are more likely to be called *partial-order* planners. In retrospect, the key idea in NOAH and Nonlin is not the partial order, but the idea of searching through a space of partial plans. (Recently, a number of papers have been published that attempt to formalize the search through partial plans, including [4,17,60].)

One important research strand in all this was the idea of *goal regression*. Given a goal that must be achieved at a point in a partial plan, what must be true before a previous action for it to become true at the right point? For example, suppose a partial plan contains the steps *Drive truck 3 to Smithville*, and *Put load 15 into warehouse A in Smithville*, and suppose that a precondition of the latter step is that load 15 be in Smithville. Regressing the precondition across the truck-driving step yields the new goal: *Either load 15 is on truck 3 or it is already in Smithville*, which must be true before the truck is driven to Smithville if that step is to result in load 15 being in the right place at the right time. This idea was first articulated in the field of program analysis and synthesis [14]. It was applied to planning by Waldinger [57] and Warren [58] in the mid-seventies, and formalized by Pednault in the eighties [38]. Their systems searched a space of partial plans that are totally ordered throughout. However, total ordering does not prevent the insertion of new steps (e.g., *Put load 15 onto truck 3*) between existing steps.

By coincidence, systems that made regression explicit were treated with more theoretical rigor than the "nonlinear" systems. As often happens in AI, theoretical rigor seemed inversely proportional to practical applicability. Descendants of Nonlin and NOAH, especially Wilkins' SIPE system [59], were attempts to build software systems that could be applied to real problems. So they included lots of tools for handling user interaction, knowledge representation, inference, and so on. It's usually not possible to say anything formal about a system with this degree of "programmability". Meanwhile, the work in regression produced a series of theorems about search techniques, such as proofs that these techniques are *complete*, that is, capable in principle of solving all classical-planning problems [30,35,44]. However, these theorems did not seem to have much bearing on the behavior of practical systems.

The disillusionment with the gap between theory and practice led to the exploration of a wide variety of new approaches that were aimed at extending the classical framework. In addition, an influential paper by Chapman [9] was published that attempted to synthesize a number of existing approaches, and to show that classical planning problems could be undecidable in many situations. These factors led to the exploration of a number of new approaches that attempted to go beyond the classical framework. Techniques included the use of memory-based approaches [3,25], parallel searches [26], time maps [12], simulation and debugging [50], localization [31] and plan reuse [29]. Other work attempted to repudiate the classical approach, and to work without explicit goal-based planning (see Section 2). People in the field talked about the "death of the classical planning framework", and the planning community flourished as exciting new approaches were debated.

In the late nineteen-eighties, while these hundred flowers were blooming, there was a sudden resurgence of interest in classical planning when McAllester and Rosenblitt published their paper [34] proving the completeness of a partial-order planning algorithm. (They gave no name to their algorithm, but it is now called SNLP.) The paper had a big impact because it was unusually elegant. It can be viewed as the extraction of the essence of partial-order planners like SIPE and Nonlin, or at least the part of those systems that manages the partial orders. McAllester and Rosenblitt's algorithm uses only a basic STRIPS-style representation of actions. The output of the algorithm is a totally ordered sequence of actions, but it produces them by working through a search space of *partial plans,* each represented as a collection of four things:

- a partially ordered set of *steps*;
- a set of *precondition goals* associated with each step, which were conditions to be made true before that step in every totally ordered completion of the partial plan;
- a set of *causal links* that commit one step to achieving a precondition of another;
- a set of *separation links* that commit a step to be ordered so that it cannot interfere with a causal link.

The operators in this search space add steps and links until a plan is reached that has no unachieved preconditions.

The algorithm is clear, and it's provably complete. It's even "systematic", which means that the same partial plan is not encountered in two different parts of the search space, [1] although there has been controversy over whether that makes any difference in practice. (See **Kambhampati, Knoblock and Yang.**) But the tremendous influence of the paper has been due mainly to its clarity, which enabled other researchers to build on it in a way that hadn't happened with earlier, more complex systems. The SNLP algorithm could be implemented easily, and it was simple (if not easy) to extend it in various directions, relaxing the assumption that actions' effects were context-independent [40] or the assumption that effects are instantaneous [39]; or allowing the choice of action to depend on run-time tests [41], allowing the planner to start from a nonempty base plan, and adding actions with probabilistic effects (**Kushmerick, Hanks and Weld**).

Unfortunately, there is as yet little evidence that all these theoretical results will have practical impact. Even the basic SNLP algorithm is exponential in practice. This is not surprising, given the results of **Erol, Nau and Subrahmanian** that almost all variants of the classical planning problem are NP-complete or worse. The result stems from the ease with which arbitrary deductive problems can be mapped into planning problems.

[1] "SNLP" stands for "Systematic NonLinear Planner".

The initial hopes that a general-purpose planner might yet be more specialized, and hence more efficient, than a general-purpose theorem prover now seem naive.

2. Dynamic world planning

As mentioned earlier, during the mid nineteen-eighties, many researchers were becoming dissatisfied with the classical assumptions about planning. One often-cited argument was that the field had originated in attempts to find plans for robots, but, in fact, robots do not satisfy the classical assumptions at all well. Because robots must survive in environments that are changing and imperfectly known, they must constantly sense their surroundings and react to what they perceive. In these conditions, it doesn't seem reasonable to model behavior as a predictable sequence of actions. In fact, a whole new breed of robots that repudiated the notion of symbolic planning were being developed at MIT [8]. In the mid-1980s, a number of approaches to planning in dynamically changing worlds were explored, and papers by Agre and Chapman [1], Georgeff and Lansky [22], Rosenschein and Kaelbling [28], Sanborn and Hendler [48], Schoppers [49], Firby [20] and others developed models of planning agents that could function in dynamically changing environments. Oversimplifying somewhat, one can look at all these approaches as attempting to create agents that could react by direct coupling of sensing (stimulus) to effecting (response). In most of this work the design was carried out by humans. The more complex the behaviors got, the harder it became to design them automatically.

Around the same time, another aspect of dynamicity was also being explored. Classically it had been assumed that a planner had as much time as it required to find a plan. There are circumstances in which taking too much time to plan will reduce the value of whatever plan is found. For example, if an industrial robot must make a plan to manipulate some objects that have just appeared on a moving conveyor belt, then the best plan in the world will be worthless if the objects have moved out of reach by the time it is constructed. Hence the problem arises of trying to find the optimal amount of time t to plan, before adopting the best plan found. This problem makes sense only if the planning algorithm can be stopped after any time interval and asked for the best plan it has discovered so far; and if its best plan changes fairly often. An algorithm with these properties is called an *anytime* algorithm [11]. If these circumstances obtain, then the optimal planning time is the t that maximizes $V(t) - C(t)$, where $V(t)$ is the expected value of the plan generated after t and $C(t)$ is the cost of delaying execution by t [7,45]. The study of planning under such temporal constraints is sometimes called *deliberation scheduling.*

A trend which has emerged more recently is the need to couple the strengths of classical planning (the ability to project into the future) with those of reacting (the ability to handle dynamicity and unexpected events). This has led to a variety of "multi-level" planning approaches, several of which have been shown to be useful with simulated and/or real robots [21,33,53]. This is an important avenue of current research, and we return to it in Section 5.

3. Special-purpose planners

Another tack which has been proposed for dealing with the complexity of general-purpose planning is to specialize the domain still further, and to try to exploit restrictions that may arise. This can be taken to an extreme in "domain-specific" planning systems, which attempt in expert-systems-like fashion to solve particular planning problems in very particular domains. Such approaches may be extremely efficient, and even commercially useful, but tend to be somewhat brittle with respect to transitioning to other planning applications. However, several limited domain areas have been shown to cut across a number of important applications, and have thus become the focus of much recent planning work.

One such special case is that of *motion planning,* the problem of finding a path from one point to another through a complex region in a metric space, typically of 2 to 10 dimensions [32]. This problem may sound esoteric, but such high-dimension spaces occur routinely in robot planning, where there is one dimension per degree of freedom of the robot. It's interesting to think about the relationship of this problem to classical planning. It obeys the classical assumptions (e.g., the assumption that the world is passive and perfectly known), but would be tricky to translate into STRIPS-style addlists and deletelists. The translation would be messy, and would probably entail an exponential blowup in the number of action-definition rules. There's no reason to think that a classical planner would get far in solving the resulting problem. And, in fact, research on this problem has proceeded entirely independent of research on classical planning.

A similar situation exists with respect to another research area that is of even more importance than robot planning, to wit, scheduling, which we defined at the beginning as finding a good order to perform a series of known tasks. Scheduling problems arise repeatedly in industry. Even slight improvements in the quality of schedules can save millions of dollars in wasted time.

Scheduling problems come in many different forms. They differ in the ways that tasks seize or consume resources. For example, in job-shop problems a task will require a machine, which it releases at the end, while in transportation problems, fuel can be consumed at a rate independent of the rate at which it is replenished. Problems also differ in the kind of ordering constraints they allow for and they differ in how the free the scheduler is to permute steps (e.g., if each task must be executed in a different location, then permuting them changes the total travel time of the schedule). Because of all this variety, it is impossible to devise a single general-purpose scheduling algorithm that works well in all cases. Each problem must be approached on its own, and its solution almost always requires the use of heuristics. In short, it's an excellent field for the application of tools from the AI toolkit.

Scheduling and motion planning have one thing in common: they are intractable, but they are "less intractable" than the full planning problem. Motion planning is exponential in the dimensionality of the space being moved through [42]. If you picture the space discretized into an n-dimensional grid of k cells in each dimension, then the intuition is that the number of grid cells to explore is k^n. And, indeed, for even $n = 3$ the problem is computationally taxing. However, it is often reasonable to assume that n is fixed, as it will be for a particular robot, and concentrate on finding ways of solving as many problems as possible as the shape of the traversable region is allowed to vary. In scheduling, the reasons for relative optimism are different. It is usually fairly easy to find a feasible schedule, that is, one that does not violate any ordering or resource-bound constraints. Then one can focus on ways to improve it. It is not necessary to get all the way to optimality in order for the effort to be worthwhile.

As with motion planning, it would be a mistake to conclude anything about the relative importance of scheduling and planning from the ratio of the numbers of papers in this issue. Research has been going on in this area for decades, usually under the label "operations research". Lamentably, this label has often kept the AI community from talking much to the OR community. The idea has grown up that there is an "AI approach" to problems that is different from the "OR approach", or, for that matter, the "robot-motion-planning approach". In practice, this kind of compartmentalization has ensured that people working in AI focus on problems defined at an unrealistic level of generality.

4. What is in this issue

This special issue of *Artificial Intelligence* should make it clear how diverse the current field of AI planning is. Most current research directions are represented by at least one of the papers in this collection. In addition, we also see how each of the three main planning thrust areas discussed above are being explored in current work.

Our definition of planning—as design of behavior—is intentionally broad. Even within the classical framework, there is room for disagreement about what a plan is. In this framework, the output of the planning process is a sequence of actions to perform, but the intermediate states of the process can be of many different sorts. In the partial-order-planning paradigm that we discussed in Section 1, planning is implemented as a search through a space of partially constrained plans.

However, it has recently been realized that one of the biggest problems in planning is search control. As discussed in Section 1, planning researchers have developed some elegant theorems about the structure and completeness of search spaces. What we don't have is ways of navigating through those spaces efficiently. It could be that researchers are discouraged by the mostly negative results that have come from complexity theory. The paper by **Erol, Nau and Subrahmanian** shows that almost all classical planning problems are intractable. It is often argued that the inefficiency of planning can be controlled by making use of stored solutions to previous planning problems. This approach is called *memory-based* or *case-based* planning. But it's hard to show formally that using old cases will help. In particular, the paper by **Nebel and Koehler** argues that memory-based planning is doomed because in the worst case a planner saves nothing by starting from a previous solution instead of from scratch. Additionally, the paper by **Bäckström** proves that various planning action-definition formalisms are all equally expressive, and nothing is to be gained by using one instead of another.

These results are all proven by finding polynomial-time reductions between problems. If you want a system that solves all planning problems of a certain class C, then you have to come to grips with the fact that some NP-complete problem can be reduced to C, or engage in sustained denial. The latter seems to be the course that most researchers have chosen. It would be more interesting to try harder to find cases where C can be constrained to a subset that is empirically if not theoretically tractable.

In order to judge whether progress is being made in this direction, we need to start measuring the performance of our planners more systematically. Two papers in our collection address this issue. The paper by **Kambhampti, Knoblock and Yang** takes an important step towards such an evaluation. They define a partial plan syntactically, as consisting of a graph of steps and other data structures. This allows them to carefully vary a family of partial-order planners along several dimensions, and to measure the impact on performance in various domains. This work ignores search control, and focuses instead on issues such as systematicity of the search space. The paper by **Howe and Cohen** presents the use of statistical analysis techniques for finding patterns in the behavior of failure-recovery modules that are called during plan execution.

One problem with much work in formalizing partial-order planning is that it does not allow for a partial plan to be carried out; there are no execution semantics for any kind of plan but the kind that is produced in the end, a totally ordered set of steps. The only meaning given to a partial plan is the set of all total plans that could be derived from it by further planning decisions. The paper by **Ginsberg** begins to address that deficit, by proposing a new kind of partial plan, which is compatible with all action sequences that include it, except for a set of sequences that has "measure 0" in a sense that the paper defines formally. This may lead to a new kind of planner, and the paper makes a promising start in that direction.

Yet another novel planning formalism is presented in the paper by **Saffiotti, Konolige and Ruspini**, which redefines the semantics of reactive plans in terms of a "fuzzy-style" multiple-valued logic. The semantics provides an alternative way of defining partial plans and their composition; a detailed study of the relationship between this approach and **Ginsberg**'s might be revealing. Saffiotti et al. provide a planning algorithm that uses classical backward-chaining techniques to build decidedly nonclassical plans.

In the area of nonclassical planning, two papers are included that discuss issues involved in reactive behaviors and deliberation scheduling—**Levinson** and **Dean, Kaelbling, Kirman and Nicholson**. Both papers build on a behavior model due to Drummond and Bresina. [15] They start with a "default" plan executor that can make fast, local decisions about behavior based on the current world state. The planner improves the executor's behavior by generating possible behavior traces in advance, in order to detect looming disasters and opportunities. This process of reasoning about future execution is called *plan projection.* The more time the planner has, the more traces it can project, thus contributing to the "anytime" property. **Levinson** presents a nondeterministic formalism for expressing behaviors, that can be used for both projection and execution. **Dean et al.** present an approach based on the theory of policy generation and evaluation. The term "policy" comes from the theory of Markov decision models, and refers to plans that consist of rules of the form *world state*$\longrightarrow$*action*, thereby specifying what action to take for each state an agent finds itself in.

In the area of special-purpose planners, we present papers from two of the most important current areas: motion planning and scheduling. The paper by **Lazanas and**

Latombe studies the problem of moving a robot through a two-dimensional space with many known landmarks. As we said before, you may be tempted to conclude from the fact that this is the only paper in this issue on motion planning that it is a relatively unimportant and undeveloped branch of the field. In fact, it is a huge area, which has attracted considerably more attention than the general-purpose planning algorithms we have described above. The reason for the attention is that practical motion-planning algorithms exist, and further improvements are urgently needed for high-level robot control. But "AI people" don't talk much to "motion-planning people", and so they are underrepresented in this issue.

Two papers in this issue are concerned with scheduling: **Sadeh, Sycara and Xiong**, which deals with finding feasible schedules, and **Miyashita and Sycara**, which deals with improving schedules. **Miyashita and Sycara** use case-based learning techniques to improve performance in the search for optimal schedules. **Sadeh, Sycara, and Xiong** test backtracking heuristics that work particularly well for the sort of constraint-satisfaction problems that arise in finding feasible schedules.

5. What is not in this issue: current and future directions

One of the drawbacks of a special journal issue is that the papers, to be journal quality, don't include the many experimental approaches to a field that are just starting to emerge—what's in the journal will necessarily be several years behind what is being presented at workshops and conferences in the coming months. In this section, we want to briefly describe some of what we see as promising directions emerging from current work. (We apologize in advance for the fact that we cannot cite all relevant references in each of these areas)

Transformational planning: Most theoretically elegant planning algorithms are based on the idea of plan *refinement,* in which every step through the search space of candidate partial plans takes the planner from a more abstract partial plan to a more concrete one. That is, each step can add steps and constraints, but can't remove or alter previous commitments. Of course, backtracking away from blind alleys in this space will require undoing past decisions, and it often takes a lot of backtracking to get from one plan in the space to another. It would seem that more general classes of plan transformations deserve some study. Practical planners such as SIPE [59] and O-Plan [10] have allowed more general plan revisions, and there have been several other papers exploring the possibilities [6,25,50,61]. However, there is as yet nothing like the elegant theoretical framework of refinement planners.

Planning analysis: Like many other parts of the field, planning research is further confused by the methodological muddle that haunts AI. Complexity analysis is giving way to empirical evaluation, but in a field as diverse as planning, it's not clear where such research may go. The papers by **Kambhampti, Knoblock and Yang** and **Howe and Cohen** are promising first steps. The field is split as to what defines interesting planning problems, what approaches to them can be evaluated in what ways, and what analytic techniques (empirical and theoretical) are most appropriate for making qualitative comparisons between planning approaches. Not only is this important for making scientific advances in this important field, but the external pressure for evaluation (from funding agents in particular) makes it clear that this area must be addressed.

Case- and reuse-based approaches: **Nebel and Koehler** argue that case-based approaches to planning have worse worst-case performance than planning from scratch. However, **Miyashita and Sycara** present a case-based reasoner that works well for realistic scheduling problems. To add to the confusion, work in plan "reuse" (cf. [29]) has been differentiated from work in case-based planning (cf. [24]) and is evolving into a separate literature. This muddle over how to use memory is emerging in many parts of AI, but appears to be particularly critical in planning, where the need for greater efficiency in complex problems is particularly acute. Systems that integrate large case-bases and well-understood reuse strategies are clearly called for.

Learning to plan: There has been much research recently into the use of automated learning techniques in the domain of planning—both classical [2, 56] and dynamic [13]. The influences of learning work can be seen in two papers in this issue where **Dean, Kaelbling, Kirman and Nicholson** and **Levinson** describe approaches which are influenced by techniques from reinforcement learning [5], but which are applied to making inferences about a search space the planner already knows about, not to acquiring new facts about the world. New work in this area needs to focus on integrating stronger learning models with more complex planning domains, and in bridging gaps between learning mechanisms and the case- and reuse-based planners described above.

Real-time planning: A related area to deliberation scheduling is that of planning in real-time domains. In particular, planners must be able to function in domains where "hard" real-time boundaries must be realized (that is, where being fast is not enough - the planner must be provably able to function within specifically bounded time limits). In such domains, planners must be able to continually function, without getting "out of sync" with the world around them. The growing need for models and theories of such systems is discussed in [36]. Simply speeding up existing systems, while itself an important area, is not adequate—as computers are being placed in more and more time critical domains (air-traffic control, nuclear plants, military systems, etc.) we must be able to demonstrate that the systems have provable properties in their decision making.

Bridging from reaction to planning: As mentioned previously, a trend which has recently emerged is the use of multi-level planners to integrate the strengths of classical planning with those of reaction-based systems. The ability to both react to the present and predict the future is critical to many complex problems including realistic mobile robotic systems, intelligent information-filtering agents and controllers for complex physical systems. This work needs to provide a continuum from very low-level planning (such as the motion planning work of **Levinson**) to the complex reasoning and projection described in **Ginsberg**. While there are many current systems being explored across the planning continuum, surprisingly few are attempting to integrate the many planning results, or to propose new paradigms that span the realm of planning problems.

Real cognitive planning models: In addition to handling more complex applications, better planning models are needed if we are to be able to say anything interesting about human (and even animal) planning. Long-range planning, as has been pointed out by psychologists, philosophers, and others, is like language—it is one of the few human behaviors that seems to vary in kind, not just complexity, from that observed in most animal species. Recent work is attempting to explore the relation of current planning theories with modern psychology—for example, a recent paper by Grafman, a neuropsychologist, and Spector, an AI scientist [52] overviews some of the synergies that can result from investigating the strong relationship between these fields.[2] However, much more needs to be done. Human planning appears to include significant components of memory and analogy use, fast future projections, "compilation" of planned behaviors into learned procedures, and mechanisms for rejecting inappropriate current actions in favor of other, often better, alternatives.

6. Final words of wisdom

We'll wind up with a brief, and admittedly biased, assessment of the limitations of the planning field today and where it needs to go. In short, we feel that there is a lot of exciting work going on, and it is in a suitably diverse set of directions, but that it is not clear whether any useful goal is in sight. There are many areas that need more exploration and a number of breakthroughs must emerge before the tachnology becomes a serious one for large-scale problem solving. While we acknowledge that planning technology has had a modest impact on realistic applications, this has mainly because of the success of special-purpose algorithms, especially schedulers.

One major problem is that the field is split between work on elegant, impractical algorithms, and complex, ad hoc, practical programs. In almost every area of AI, including vision, learning, and even natural language, there has been a progression from general-purpose approaches to approaches that actually do something useful on a narrowly defined set of problems. This progression is typically *not* a progression from theoretical elegance to untrammeled hacking, or from "neat" to "scruffy", quite the contrary; narrowing the scope of a field often allows the use of more sophisticated formal tools. (One thinks of Bayesian statistics in vision, learnability theory in learning, and hidden Markov models for language processing.) But in planning we have the opposite situation. When a planner actually solves an interesting class of problems, usually the most one can say is "we programmed it that way", or, "we got lucky".

Part of the reason for this state of affairs is that work on general-purpose planners has primarily occurred at some distance from real problems, and it is not clear whether that work will ever produce algorithms that work well enough on problems of realistic size. While the purpose of planning should not be *only* to create applications, it is clear that realistic problems whether viewed from an industrial or a cognitive perspective require scaling planning approaches way beyond what they can do today.

[2] See also [27].

Unfortunately, specializing and/or scaling a planning domain usually yields a set of problems that involve a lot of reasoning techniques from other fields. Motion planning involves computational geometry. Transportation scheduling involves combinatorial optimization. If someone signed on to do general-purpose planning, he or she may feel it is conceding defeat to put planning techniques on the back burner and go learn something boring like operations research.

One possible solution to this methodological conundrum is to view general-purpose planning as providing an architectural framework for combining results from more specialized systems. That is, the general-purpose system provides a common ground for talking about plans and transformations on plans, and thereby provides a protocol for specialized reasoning algorithms to plug into. Another possibility is that some other view of planning, one that encompasses more of the field, for example coupling real robotics work with AI approaches to anytime algorithms and dynamic planning, will prove to yield more powerful formalisms that span a wider range of planning-related problems. This would allow a recoupling of theory and practice, and could lead in exciting new directions.

However, we're pessimistically forced to conclude that another solution is somewhat more likely. It may be inevitable for the field of planning to split into even smaller subfields, each with its own domain of interest (manufacturing, deliberation scheduling, logistics planning, etc.). After all, there may not be much in common between designing the behavior of a robot and designing the behavior of a military logistics organization. It would be a pity to see this happen, but if what is gained is a set of elegant and powerful theories coupled with useful implementations replacing the current elegant but weak theories coupled with toy systems, then maybe it will be worth it.

Acknowledgements

Dr. McDermott's research is supported in part by a grant from ARPA, administered by ONR, number N00014-93-1-1235. Dr. Hendler's research was supported in part by grants from NSF(IRI-9306580), ONR (N00014-J-91-1451), AFOSR (F49620-93-1-0065), the ARPA/Rome Laboratory Planning Initiative (F30602-93-C-0039), and the ARPA I3 Initiative (N00014-94-10907) and by ARI (MDA-903-92-R-0035, subcontract through Microelectronics and Design, Inc.) Dr. Hendler is also affiliated with the UM Institute for Systems Research (NSF Grant EEC 94-02384).

References

[1] P.E. Agre and D. Chapman, Pengi: an implementation of a theory of activity, in: *Proceedings AAAI-87*, Seattle, WA (1987) 268–272.

[2] J.A. Allen and P. Langley, Integrating memory and search in planning, in: K. Sycara, ed., *Proceedings of the Workshop on Innovative Approaches to Planning, Scheduling, and Control* (Morgan Kaufmann, San Mateo, CA, 1990) 301–312.

[3] R. Alterman, Adaptive planning, *Cognitive Sci.* **12** (1988) 393–422.

[4] A. Barrett and D.S. Weld, Task-decomposition via plan parsing, in: *Proceedings AAAI-94*, Seattle, WA (1994).

[5] A.G. Barto, R.S. Sutton and P.S. Brouwer, Associative search network: a reinforcement learning associative memory, *Biol. Cybern.* **40** (3) (1981) 201-211.

[6] M. Beetz and D. McDermott, Revising failed plans during their execution, in: K.J. Hammond, ed., *Proceedings Second International Conference on AI Planning Systems.* (Morgan Kaufmann, San Mateo, CA, 1994).

[7] M. Boddy and T.L. Dean, Solving time-dependent planning problems, in: *Proceedings IJCAI-89*, Detroit, MI (1989) 979-984.

[8] R.A. Brooks, A robust layered control system for a mobile robot, *IEEE J. Rob. Automation* **2** (1986) 14-23.

[9] D. Chapman, Planning for conjunctive goals, Technical Report 802, MIT AI Lab, Cambridge, MA (1985); also: *Artif. Intell.* **32** (1987) 333-377.

[10] K. Currie and A. Tate, O-plan: the open planning architecture, *Artif. Intell.* **52** (1) (1991) 49-86.

[11] T.L. Dean and M. Boddy, An analysis of time-dependent planning, in: *Proceedings AAAI-88*, St. Paul, MN (1988) 49-54.

[12] T.L. Dean and D. McDermott, Temporal data base management, *Artif. Intell.* **32** (1) (1987) 1-55.

[13] G.F. DeJong, Explanation-based control: an approach to reactive planning in continuous domains, in: K. Sycara, ed., *Proceedings of the Workshop on Innovative Approaches to Planning, Scheduling, and Control* (Morgan Kaufmann, San Mateo, CA, 1990) 325-336.

[14] E.W. Dijkstra, *Discipline of Programming* (Prentice-Hall, Englewood Cliffs, NJ, 1976).

[15] M. Drummond and J. Bresina, Anytime synthetic projection: maximizing the probability of goal satisfaction, in: *Proceedings AAAI-90*, Boston, MA (1990) 138-144.

[16] G.W. Ernst and A. Newell, *GPS: A Case Study in Generality and Problem Solving* (Academic Press, New York, 1969).

[17] K. Erol, J. Hendler and D.S. Nau, HTN planning: complexity and expressivity, in: *Proceedings AAAI-94*, Seattle, WA (1994).

[18] R.E. Fikes, P.E. Hart and N.J. Nilsson, Learning and executing generalized robot plans, *Artif. Intell.* **3** (4) (1972) 349-371.

[19] R.E. Fikes and N.J. Nilsson, STRIPS: a new approach to the application of theorem proving to problem solving, *Artif. Intell.* **2** (1971) 189-208.

[20] R.J. Firby, An investigation into reactive planning in complex domains, in: *Proceedings AAAI-87*, Seattle, WA (1987).

[21] E. Gat, Reliable goal-directed reactive control of autonomous mobile robots, Doctoral Dissertation (revision 1.1), Virginia Polytechnic Institute and State University (1991).

[22] M.P. Georgeff and A. Lansky, Procedural knowledge, in: *Proceedings IEEE Special Issue on Knowledge Representation* **74** (10) (1986) 1383-1398.

[23] C. Green, The application of theorem proving to question-answering systems, Ph.D. Thesis (1969).

[24] K.J. Hammond, Chef: a model of case-based planning, in: *Proceedings AAAI-86*, Philadelphia, PA (1986).

[25] K.J. Hammond, Explaining and repairing plans that fail, *Artif. Intell.* **45** (1-2) (1990) 173-228.

[26] J. Hendler, *Marker-passing and Problem Solving: A Spreading Activation Approach to Improved Choice in Planning* (Lawrence Erlbaum, Hillsdale, NJ, 1987).

[27] J. Hendler, Types of planning—can artificial intelligence yield insights into prefrontal function?, in: *The Frontal Lobes—Annals of the New York Academy of Science* (in press).

[28] L.P. Kaelbling and S.J. Rosenschein, Action and planning in embedded agents, in: Patti Maes, ed., *New Architectures for Autonomous Agents: Task-level Decomposition and Emergent Functionality* (MIT Press, Cambridge, MA, 1990) 35-48.

[29] S. Kambhampati and J.A. Hendler, A validation-structure-based theory of plan modification and reuse, *Artif. Intell.* **55** (1992) 193-258.

[30] H.A. Kautz, A first-order dynamic logic for planning, Technical Report 144, University of Toronto (1982).

[31] A. Lansky, (1991) "Localized Event-Based Reasoning For Multiagent Domains, *Comput. Intell.* **4** (4) (1991).

[32] J.-C. Latombe, *Robot Motion Planning* (Kluwer Academic Publishers, Boston, MA, 1991).

[33] D. Lyons and A. Hendriks, Testing incremental adaptation, in: K.J. Hammond, ed., *Proceedings Second International Conference on AI Planning Systems.* (Morgan Kaufmann, San Mateo, CA, 1994).

[34] D. McAllester and D. Rosenblitt, Systematic nonlinear planning, in: *Proceedings AAAI-91*, Anaheim, CA (1991) 634-639.

[35] D. McDermott, Regression planning, *Int. J. Intell. Syst.* **6** (4) (1991) 357–416.

[36] D. Musliner, A. Agrawala, E. Durfee and J. Strosnider, The challenges of real-time AI, *IEEE Computer* **28** (1) (1995).

[37] A. Newell and H. Simon, GPS: a program that simulates human thought, in: *Lernende Automaten* (R. Oldenbourg KG) 279–293; reprinted in: E.A. Feigenbaum and J. Feldman, eds., *Computers and Thought* (1963).

[38] E.P.D. Pednault, ADL: exploring the middle ground between STRIPS and the situation calculus, in: *Proceedings First International Conference on Principles of Knowledge Representation and Reasoning*, Toronto, Ont. (1989) 324–332.

[39] J.S. Penberthy, Planning with continuous change, Technical Report 93-12-01, University of Washington, Department of Computer Science and Engineering, Seattle, WA (1993).

[40] J.S. Penberthy and D.S. Weld, UCPOP: a sound, complete, partial order planner for ADL, in: *Proceedings Third International Conference on Principles of Knowledge Representation and Reasoning*, Boston, MA (1992).

[41] M. Peot and D. Smith, Conditional nonlinear planning, in: J. Hendler, ed., *Proceedings First International Conference on AI Planning Systems* (Morgan Kaufmann, San Mateo, CA, 1992) 189–197.

[42] J. Reif, Complexity of the Mover's Problem and generalizations, in: *20th Proceedings IEE Symposium on Foundations of Computer Science* (1979) 421–427.

[43] E. Rich and K. Knight, *Artificial Intelligence* (McGraw-Hill, New York, 2nd ed., 1991).

[44] S.J. Rosenschein, Plan synthesis: a logical perspective, in: *Proceedings IJCAI*, Vancouver, BC (1981) 331-337.

[45] S. Russell and E. Wefald, On optimal game-tree search using rational meta-reasoning, in: *Proceedings IJCAI-89*, Detroit, MI (1989) 334–340.

[46] E.D. Sacerdoti, Planning in a hierarchy of abstraction spaces, in: *Advance Papers IJCAI-73*, Stanford, CA (1973).

[47] E.D. Sacerdoti, *A Structure for Plans and Behavior* (American Elsevier, New York, 1977).

[48] J. Sanborn and J. Hendler, A model of reaction for planning in dynamic domains, *Int. J. AI Eng.* **3** (2) (1988).

[49] M. Schoppers, Universal plans for reactive robots in unpredictable environments, in: *Proceedings IJCAI-87*, Milan, Italy (1987) 1039–1046.

[50] R.G. Simmons, The roles of associational and causal reasoning in problem solving, *Artif. Intell.* **53** (2-3) (1992) 159-207.

[51] L. Siklossy and J. Dreussi, An efficient robot planner that generates its own procedures, in: *Proceedings IJCAI-75*, Tblisi, Georgia (1975).

[52] L. Spector and J. Grafman, Planning, neuropsychology, and artificial intelligence: cross-fertilization, in: F. Boller and J. Grafman, eds., *Hanbbook of Neuropsychology* **9** (Elsevier, North-Holland, Amsterdam, 1994).

[53] L. Spector and J. Hendler, Planning and control across supervenient levels of representation, *Int. J. Intell. Cooperative Information Syst.* **1** (3-4) (1992).

[54] G.A. Sussman, A computational model of skill acquisition, MIT AI Lab. Memo No. AI-TR-297, AI Laboratory, MIT, Cambridge (1973).

[55] A. Tate, Generating project networks, in: *Proceedings IJCAI-77*, Cambridge, MA (1977) 888-893.

[56] M. Veloso and J. Carbonell, Derivational analogy in PRODIGY: automating case acquisition, storage, and utilization, *Mach. Learn.* **10** (1993) 249–278.

[57] R. Waldinger, Achieving several goals simultaneously, in: E. Elcock and D. Michie, eds., *Machine Intelligence* **8** (Ellis Horwood, Chichester, 1977) 94–136; also in N.J. Nilsson and B. Webber, eds., *Readings in Artificial Intelligence* (Tioga, Palo Alto, CA, 1981).

[58] D.H.D. Warren, Warplan: a system for generating plans, Technical Report 76, University of Edinburgh, Department of Computational Logic (1974).

[59] D. Wilkins, *Practical Planning: Extending the Classical AI Planning Paradigm* (Morgan Kaufmann, San Mateo, CA, 1988).

[60] R. Young, M. Pollack and J. Moore, Decomposition and causality in partial-order planning, in: K.J. Hammond, ed., *Proceedings Second International Conference on AI Planning Systems.* (Morgan Kaufmann, San Mateo, CA, 1994).

[61] M. Zweben, E. Davis, B. Daun and M. Deale, Scheduling and rescheduling with iterative repair, *IEEE Trans. Syst. Man Cybern.* **23** (6) (1993) 1588–1597.