The Centrality of Autonomous Agents in Theories of Action Under Uncertainty

Paul R. Cohen and David S. Day
Department of Computer and Information Science University of Massachusetts

ABSTRACT

In this paper we discuss a class of tasks in which to study planning under uncertainty. We analyze the interaction between autonomous agents and uncertain environments, and review the artificial intelligence literature on planning from this perspective. Case studies from our own research are presented to illustrate issues in planning under uncertainty and methods for studying these issues.

KEYWORDS: ***planning, real-time planning, control, iterative decision making, autonomous agents, artificial intelligence***

INTRODUCTION

Reasoning under uncertainty has two aspects. One is to assess the most likely states of the world, and the other is to act on those assessments. The former is often called *judgment* and the latter *decision making*. Judgment is the primary focus of research on reasoning under uncertainty in artificial intelligence (AI). Although many AI systems make decisions, those that serve as examples of research on uncertainty typically do not. Instead, the literature on uncertainty in AI is concerned almost exclusively with a single aspect of a single generic task: combining evidence in interpretation tasks. Expert systems for medical diagnosis serve as the canonical AI systems in this research. Other aspects of interpretation tasks that do involve decisions, such as deciding which evidence to acquire and deciding how to treat a patient given a diagnosis, have been largely ignored. Other generic tasks and domains, such as planning, design, robotics,

Address correspondence to Paul Cohen, Computer Science Department, University of Massachusetts, Amherst, Massachusetts, 01003.

International Journal of Approximate Reasoning 1988; 2:303–326

52 Vanderbilt Ave., New York, NY 10017 0888-613X/88/$3.50

and process control, have also been neglected. But AI is increasingly concerned with decision making in such tasks and domains and is much less concerned with judgment in simple diagnosis. Thus, the research published in, say, the proceedings of the AAAI Workshops on Uncertainty and Artificial Intelligence is largely irrelevant to many current AI research problems, even though these problems involve considerable uncertainty.[1]

This paper describes a class of problems that concern how agents act in uncertain environments. Lately, these have been called *planning problems* and the agents *planners* (e.g., McDermott [5] and DARPA [6]). We are writing this paper to urge researchers in the uncertainty community to expand their focus to include planning problems. We advocate this for several reasons. First, although uncertainty is the most salient characteristic of planning problems, current research in the uncertainty community tells us nothing about how to design planners. Second, much of the literature on planning comes from the logicist community in AI (e.g., Georgeff and Lansky [7]) and so emphasizes nonmonotonic reasoning over probabilistic approaches to uncertainty. But probabilities (if you can get them) are better suited than assumptions to the task of selecting actions, because they can be combined with utilities (if you can get *them*) and ranked.

Third, we believe that too much energy is devoted to making ever-finer distinctions between methods for combining evidence, just as symmetric efforts are devoted to showing that these methods in fact mathematically subsume one another or can be incorporated one within the other. These debates ignore a fundamental question: For what *purpose* are we combining evidence? Interpretation tasks provide the illusion of a purpose; but when a researcher from the uncertainty community says "medical diagnosis," he or she usually means "evidence combination" only, not planning the diagnosis, not deciding between treatments, and not prognosis. In contrast, planning depends on evidence combination and so provides a context for research on judgment. Planners need to interpret data from the world well enough to act; they need to know the likely outcomes of actions well enough to select among the actions; they need to know the probabilities of events beyond their control well enough to prepare for them.

---

[1] For example, of 51 papers published in the *Proceedings* of the Third Annual Workshop on Uncertainty in Artificial Intelligence, roughly one-half mentioned no application whatsoever but were clearly influenced by diagnostic expert systems, two described vision systems, and all but four of the rest described diagnostic applications, or algorithms for learning or knowledge acquisition for diagnostic applications. The remaining four are Tong and Appelbaum's discussion of the relationship between knowledge representation and evidential reasoning in information retrieval tasks [1]; Steve Hanks' discussion of the persistence problem in planning (i.e., how the passage of time affects one's belief in propositions) [2]; Cohen's description of a program and an architecture for planning diagnoses [3] (see also the fourth section of this paper); and Wellman and Heckerman's analysis of the range of tasks facing all intelligent agents in moderately complex environments [4].

How well is "well enough"? Let us not debate this in abstract interpretation tasks, but in the context of planning tasks.

Planning is concerned with the interaction of agents and their environments. Time is an important dimension of this interaction; agents may act or remain inactive, but time still passes. Agents are assumed to have goals—among them, finding out about the environment, changing it in some way, changing themselves, changing their relationships with each other and with the environment, and so on. Agents also have plans, which for now are just internal structures that dictate how agents respond to their environment. Plans may be reflexes, multiaction schemas, symbolic contingency plans, and so on. Agents sense their environments. Often, many layers of inference separate sensation from perception. Agents also adapt to their environments.

Uncertainty is the most salient characteristic of the relationship between agents and their environments. Consider some sources of uncertainty: Agents' knowledge about the environment may be inaccurate and incomplete. The space of possible futures expands combinatorially, so agents cannot foresee the outcomes of more than a few actions. An agent is typically not the only actor in an environment, and the behavior of other agents may be unpredictable. The environment itself may be unpredictable. Agents may have limited time to reason and act; to meet deadlines they may have to rely on heuristic, approximate, and thus uncertain methods (Lesser et al. [8]). Time also introduces questions about persistence; for example, how does passing time affect belief in a predicate such as "can't eat another thing" (Hanks [2])?

Time is important for another reason: Agents themselves persist and may have many opportunities to achieve their goals, so the "one-shot" view of decision making is typically not appropriate. Wellman and Heckerman [4], who introduce the term one-shot, point out that in most situations agents are not required to decide anything but can instead collect data, converse, run experiments, and so on. Decisions are rarely unrecoverable; agents can usually recover from decisions with bad outcomes at some cost.

Many tasks can be described in terms of agents interacting with their environments. Those studied in AI include robot path planning, process control, intensive care unit monitoring, learning to avoid air traffic control mishaps, planning diagnoses, and fighting forest fires. All these tasks demand judgment, that is, combining evidence to assess the current state of the world. But they also require decisions about how to act in pursuit of goals in environments that are uncertain for the reasons described above. We urge the uncertainty community to embrace these tasks because they offer much richer opportunities to study reasoning under uncertainty, especially decision making, than simple interpretation tasks.

This article discusses planning from three perspectives. The following section focuses on the design of planners, the third presents an overview of the AI planning literature in terms of these design issues, and the fourth discusses planners we have built specifically to study reasoning under uncertainty.

SELECTING ACTIONS UNDER UNCERTAINTY

How do agents select actions to achieve their goals in uncertain environments? AI and related fields offer a somewhat bewildering array of answers. Some are very general; for example, best-first search (Barr and Feigenbaum [9]) and decision analysis (Winterfeldt and Edwards [10]) dictate that agents should "do what's best," as assessed by unspecified heuristic evaluation functions and utility functions, respectively. General planning algorithms find sequences of actions, prior to their execution, to satisfy goals and constraints (e.g., Cohen and Feigenbaum [11]). In fact, this is a traditional view of planning; today, planning denotes several other methods, most of which assume some knowledge about the environment. These include case-based planning, where agents recall, modify, and execute plans from memory (Hammond [12], Sycara [13]), and reactive planning, where agents are constructed to respond automatically to changes in the environment (see later sections for details). Finally, we have seen many *control* approaches to selecting actions under uncertainty. Control strategies specify how programs should act. New control strategies are invented when programs do not behave properly (e.g., they ask questions in the wrong order or consider alternatives irrespective of their prior probabilities). Sometimes we can pry apart control strategies from their implementations, but in general this is difficult (Gruber and Cohen [14]). We have discussed the problem of selecting actions under uncertainty from the perspective of control in recent papers [3, 15, 16] and have surveyed the relevant literature [17]. Here, we do not discuss control except in the case studies section.

In this section, we describe the issues that arise when designing autonomous agents that operate in complex, real-time, dynamic environments. These issues drive our work on reasoning under uncertainty.

Internal and External Action

Are the actions taken by the agent *internal* or *external*? Intuitively, this is the distinction between thinking and acting. For example, chess requires an internal search of moves and countermoves before committing to an external action, an irrevocable move. Similarly, planning traditionally involves searching internally for an ordered sequence of actions that will achieve a goal and then executing them externally. Many AI systems take no external actions at all; for example, some natural language parsers simply take input and process it internally. Other systems' external actions are limited to requesting data; for example, the MUM system (see case studies section) is designed to ask medical questions in an appropriate order.

Balancing Internal and External Actions

What factors affect the balance between internal and external action? Should we explore the space of alternative actions before committing to one, or should we commit to the first action that seems appropriate? The answer in general depends on the costs and benefits of searching through the outcomes of external actions. This search is referred to in the planning literature as *projection*. Sometimes we cannot project, or projected outcomes may be irrelevant, or we may lack the resources to compute them. When projection is uninformative, we may simply execute an action to see what happens. A related question is: When should a problem solver anticipate the outcomes of *multiple* external actions? Since actions are typically not independent, the cost of searching the space of joint outcomes is combinatorial in the number of actions. If, in addition, we cannot predict some of the outcomes, then searching this space may not be worth the effort.

Representing the External World

How accurate is the internal representation of the external environment? We assume that an AI program's internal representation of a chessboard is accurate, at least with respect to the positions of pieces. Traditional planners made similar assumptions: The environment is precisely as we represent it and does not change except by our action, and our knowledge of these changes is complete and accurate. Clearly, planning in the real world cannot proceed on these assumptions. Much of the discussion later in this paper is about how to plan when the internal representation of the external environment is incomplete and inaccurate.

Uncertainty in a planner's world model comes from several sources, including the following:

- The planner may have an inaccurate or incomplete understanding of the principles that govern the dynamic behavior of the environment.
- The planner may have inaccurate information about the current state of the environment.
- The system may not have adequate time (or other resources) to assess the state of the environment, the outcomes of actions, or changes in the environment beyond its control.

Real-Time Constraints

Are real-time constraints placed on the agent by the environment? As agents are designed to have more autonomy, we must be concerned about the timeliness of their planning, plans, and actions. We regard real-time planning problems as falling between the extremes of adequate time to produce an excellent solution and inadequate time to produce *any* solution. Real-time planning is thus concerned with trade-offs between the quality—broadly construed—and the time requirements of a plan. We do not regard faster computers or more efficient planning algorithms as solutions to the real-time planning problem in general, though they will obviously help in specific applications. The general real-time problem starts with the premise that the available time will at some point be inadequate. It demands that our planners adapt their processing to produce the best possible solution in the available time. This has been called *approximate processing* because it implies that the solution to a problem will approach but not meet all of our goals (Lesser et al. [8]).

Constraints Between Actions

Actions are rarely independent; can dependencies between actions help select actions? Some actions interfere with others, so that one prevents another or undoes the outcome of another. Some actions are redundant, so one achieves the same state or gets the same evidence as another. If the outcomes of actions were certain, then redundancy might be inefficient; but we can also exploit redundancy to make up for uncertainty about the outcomes of actions. In addition, constraints between actions will determine the extent to which taking one action commits the agent to a particular series of actions in the future, and so emphasizes the importance of considering such constraints before prematurely committing to a single action.

Where Do Plans and Goals Come From?

Do actions come from internally generated goals and plans, or are goals and plans epiphenomena of the direct interaction of the agent with its environment? In most AI planners, goals and plans select actions. But the environment can play the same role. Thus, an agent that has no goals or plans but responds to its immediate environment will appear to have goals and plans if the environment provides regularity and continuity. From this perspective, goals and plans are not explicit structures within the agent but emerge from its interaction with its environment. An intermediate position is associated with "Simon's ant" (Simon [18]). The problem solver is assumed to have goals and plans that bias its selection of actions, but the environment also plays a greater or lesser role.[2]

[2] Simon notes that the path of an ant in a sand dune appears on a fine scale to be almost random, since the ant must respond to random obstacles. Yet it has a general direction dictated by where the ant wants to go.

Global and Local Evaluation of Actions

What is the relationship between global properties of plans and local planning decisions? Plans are generated by selecting actions, so to generate ''good'' plans one must somehow evaluate the potential component actions. It is prohibitively expensive to calculate the extended ramifications of actions, so many planners base decisions about actions on their local outcomes. But sequences of actions that look good from a local perspective are not necessarily good plans, because they may not satisfy global criteria; for example, an agent that always selects the cheapest applicable action will not find the cheapest plan if that plan requires a sequence of actions ordered by the inverse of their costs. The agent's evaluation of the local outcome of actions, which may frequently ensure low-cost plans, does not guarantee the lowest-cost plan in all cases.

Real-world plans must satisfy so many criteria that agents will not be able to generate optimal plans, that is, plans in which the outcomes of all actions look as good as possible from a global perspective. Plans should be inexpensive, flexible, and likely to succeed; they should take relatively little time to generate, require little monitoring, and so on. Since optimality is not a realistic aim, it is neither realistic nor desirable to project the global ramifications of actions. Nevertheless, agents can get into serious trouble if they do not project to some depth the ramifications of some actions. In the case studies section, we discuss the characteristics of the environment that require agents to project and, conversely, the kinds of environments in which agents can generate good (if not optimal) plans without any projection, that is, by evaluating only the local ramifications of outcomes of actions.

Adaptation

A problem-solving agent must have knowledge to act in the ways advocated in this article. As we note in the following section, general planners are weak, but powerful planners require knowledge about their environments. This raises three issues: What is learned, how is it learned, and what form does it take? To plan by projection, agents must know which actions are applicable and which outcomes are possible, given their goals and the state of the environment; and they must evaluate those outcomes by projection. Planning without projection also requires knowledge about the applicability of actions, but it may not require anything else. An agent may simply maintain a list of situation-actions pairs that tell it what to do in each state of the environment. In this case, the agent's actions are selected by its environment. Such agents are *adapted* to their environments to the extent that they have learned which actions are appropriate in which states of the environment. We believe that most agents will project in some situations and simply react in others, so both kinds of knowledge must be learned.

One advantage of reaction, as opposed to projection, is that agents can learn how to act in the absence of explicit knowledge about the dynamics of the environment. For example, one can learn to ride a bicycle—how to balance, steer, and so on—without knowing the physics of balancing and propelling the bicycle. Anecdotal and scientific evidence indicates that one should *not* ride a bicycle by projecting the ramifications of each tilt and wobble. The reason we do not need to project in these situations is that their outcomes always depend on predictable interactions of the same factors—gravity, one's angle, velocity, and so on.[3] Action sequences such as pedalling a bicycle are often called skills. In humans they tend to be automatic, that is, to require no conscious effort.[4] People work hard to develop skills in real-time environments such as driving cars, flying jets, and most sports, because these environments do not afford the opportunity to think through the implications of each action (see earlier discussion of real-time constraints).

Although most knowledge bases are built by hand, agents in complex environments will have to learn autonomously how to act. Even in a simple medical domain, we found it necessary to augment standard knowledge-acquisition mechanisms with an ability to automatically generalize the acquired knowledge to a broader range of situations (see ASK and MU case studies). AI has developed many learning mechanisms (e.g., Michalski et al. [23]) that may be appropriate for agents learning how to act.

The knowledge that agents need to select actions can take many forms, including situation–action rules, general preference rules, contingency plans, utility functions, scripts, and so on. The form of the knowledge depends on how it will be used, and in turn on the demands placed on the agent by the environment. For example, we noted above that humans adopt automatic skills in time-constrained environments; unfortunately, skills are inflexible and difficult to interrupt. Thus, agents must learn skills that run to completion before the environment changes in ways that make them inapplicable.

AN OVERVIEW OF THE PLANNING LITERATURE

AI research in planning, which has concerned itself more than any other part of AI with the problems of selecting actions to achieve goals, can be viewed from the perspective of these dimensions. Early on, researchers adopted predicate calculus as a representation language for planners and viewed planning

[3] Paradoxically, projection should be reserved for situations where the relationships between actions and outcomes are not completely predictable; if they were, then they would eventually be learned as situation–action pairs.

[4] For a discussion of the distinction between automatic and controlled behavior, see Schneider and Shiffrin [19]. Computational models of these behaviors and their interactions have been suggested by Schneider [20] and Day [21, 22].

metaphorically (and sometimes literally) as theorem proving. The initial state of the world was captured in a set of axioms, the goal state was a theorem, and the plan was a proof that the goal state could be reached from the initial state. This approach to planning is found in HACKER (Sussman [24]), GPS (Newell and Simon [25]), STRIPS (Fikes et al. [26]), INTERPLAN (Tate [27]), Waldinger's planner [28], ABSTRIPS (Sacerdoti [29]), NOAH (Sacerdoti [30]), and NONLIN (Tate [31]).

From our perspective, the most salient characteristic of these planners was their almost complete avoidance of uncertainty. These programs were crafted under the assumptions that they would have complete, accurate knowledge about the state of the environment and complete, accurate knowledge about the immediate outcomes of all actions. Actions were represented (and indexed) in terms of their immediate outcomes; for example, in the blocks world, one of the immediate outcomes of the action *(take-off x y)* is *(clear-top y)*—removing *x* from *y* clears off the top of *y*. But although these planners knew the immediate outcomes of actions, they were required to search combinatorial spaces of extended ramifications. For example, an action such as *(put-on red-block green-block)* may achieve an immediate goal but may later impede progress toward a goal that requires *green-block* to have a clear top. A better plan may involve moving *green-block* first and then putting *red-block* on it. Early planners dealt with uncertainty about the extended outcomes of actions by various forms of search. Some algorithms were more efficient than others (i.e., required less backtracking). But all assumed that, because the immediate outcomes of actions are certain, the extended ramifications could be discovered by projection, that is, by internally simulating the actions in a plan before committing to any external actions.

More recently, AI has developed planning methods that do not depend so heavily on these assumptions. One approach, which modifies the earlier planning algorithms relatively little, involves *replanning* when the environment turns out to be different than projected. For example, Wilkins' SIPE planner was very much like NOAH [30], but when discrepancies were detected between the environment and its internal representation, SIPE would efficiently modify its plan, maintaining as much of its original plan as possible (Wilkins [32]). Related replanning methods have been developed by Broverman and Croft [33].

Early planners assumed that actions were instantaneous and that their effects persisted until they were explicitly negated. However, actions take time, and the states they bring about may be ephemeral. Temporal logics and temporal planners address these issues (e.g., see McDermott [34], Allen [35], and Dean [36] for work on temporal logic, and Vere [37] and Hanks [2] for temporal planners).

Just as actions take time in real physical environments, planning and replanning themselves take time (see previous section). In real-time planning, the agent must allocate a limited resource—time—to planning, replanning, monitoring the environment, and action. Time usually has costs (e.g., in the fire-fighting domain discussed in the PLASTYC case study, later in this paper, forests are consumed while the agent plans). The balance between internal and external actions is further complicated by uncertainty about the environment: To meet time constraints, agents may begin sequences of actions before they have all the evidence they need; but if they wait, an opportunity may be lost and the evidence will be of no value. Recent work on real-time planning includes that by Durfee [38], Lesser et al. [8], Hayes-Roth and her colleagues [39–41], Korf [42], Dacus [43], Luhrs et al. [44], Herman et al. [45], Daily et al. [46], Firby and Hanks [47], Hendler and Sanborne [48], and others.

A radically different approach challenges the distinction between planning and execution, and thus the distinction between internal and external action (discussed earlier). This view holds that, by any metric, projection in uncertain environments is inefficient: Dynamic environments will never be as they are projected to be, so projection is a waste of resources. Projection involves selecting actions that are expected to be appropriate at some point in the future; planning without projection involves selecting actions based on the current, immediate environment, without explicitly considering their consequences. *Reactive planners* do not project but simply react to their environments, so the distinction between planning and execution is absent (Chapman and Agre [49, 50], Brooks [51], Firby [52]).

Reactive planning raises questions about the status of goals; planners may *appear* to be goal-directed when, in fact, they are simply responding to their environments. One does not need internal structures called goals to explain apparently intentional behavior. But we believe that intelligent agents should reason about their goals, so some goal-directed behavior will not be generated by reactive planning (McDermott [5], Dean [53]).

Reactive planners need to know *how* to respond to different situations. They recognize situations and respond appropriately, so to be adapted to complex environments, reactive planners need to recognize many situations. This requires so much knowledge that it will be impossible to build reactive planners for some environments; instead, reactive planners must learn how to respond to situations through interactions with their environments. Some AI techniques for learning concepts have been suggested or applied to the task of learning the situation–action contingencies required for reactive planning; these include connectionist learning (Barto et al. [54], Sutton and Barto [55], Jordan [56], Rumelhart and Norman [57]), knowledge acquisition and generalization (Gruber [58]; see also ASK and MUM case studies, this article), chunking (Laird et al. [59]), and production rule learning (Mitchell et al. [60], Anderson [61]).

We believe reactive planning is an extreme response to uncertainty in the environment. We agree that the value of projection depends on the certainty with which we can predict the outcomes of actions; but since this is variable, and depends on many factors, we do not agree that planners should completely forgo projection. [Others have made similar observations (Hayes-Roth [62], Dean [53], Swartout [63]).] We illustrate this later in the context of two of our own planning systems.

We can, in fact, project even if we do not know the precise outcomes of actions. Decision analysis is a form of planning by projection: When actions have uncertain outcomes, and these lead to further actions, then a combinatorial space of actions and outcomes is quickly generated (see earlier subsection "Representing the External World"). If one knows the probabilities that actions will lead to particular outcomes, and also the utilities of the outcomes, then one can find the *subjective expected utility* of actions. For example, imagine test-1 costs $10 and will accurately say whether or not a patient has disease A but says nothing about diseases B and C; and test-2 costs $50 and is diagnostic for disease B but says nothing about A or C. Which diagnostic action should we take first, test-1 or test-2? Assuming A, B, and C have equal priors, it is most efficient to do test-1 first. Decision analysis will resolve the question for the general case of unequal priors.[5]

But note that two assumptions are implicit in the example: The statement of the problem implies that the hypotheses A, B, and C are both *mutually exclusive* and *exhaustive*. Thus, if test-1 finds A, we need not do test-2; and if both tests fail to find A and B, then the answer must be C. Mutual exclusivity is a special case of conditional probability: When we say A and B are mutually exclusive we mean that the probability of disease B is conditional on the outcome of test-1 (and equivalently, our belief in A), and vice versa. Thus, in the general case, to plan a sequence of tests to find out which disease a patient has, we need to know the conditional probability of each disease given each combination of outcomes of tests for these diseases. Even if we assume that the diseases are mutually exclusive and exhaustive, and we assume that every test either confirms or disconfirms a disease (but does not provide partial support for any disease), we are still faced with a combinatorial search because there are $N!$ sequences of diagnostic actions, and because each action can have several possible outcomes.

We are not rejecting decision analysis as a technique for planning under uncertainty, only noting that its inherent combinatorics must be managed carefully (e.g., Wellman and Heckerman have proposed some approximate forms of decision analysis [64]). All planning by projection generates combinatorial spaces of plans; uncertainty about outcomes simply makes the problem worse. We expect that decision analysis can be merged with AI planning techniques (e.g., least commitment and hierarchical planning) to

---

[5] We are grateful to Professor Glenn Shafer for this example.

reduce the combinatorics of projection. For example, hierarchical planning reduces the combinatorics of projection by generating plans at successive levels of abstraction (Cohen and Feigenbaum [11]). Decision analysis might be used to select actions at each level.

CASE STUDIES IN PLANNING UNDER UNCERTAINTY

In this section we describe some of our research on planning under uncertainty. We begin with MUM, a system for planning medical workups. MUM led to a shell for building planners called MU, which in turn is the basis of a knowledge-acquisition system called ASK. We also describe a system called PLASTYC that, though not yet completed, highlights differences between reactive planning and planning with projection and also illustrates how simulators can facilitate research on planning under uncertainty.

MUM

The MUM system plans diagnostic workups of chest pain (Cohen et al. [15]). Its goal is to ask questions, request tests, and prescribe therapies in an efficient order. By efficient, we mean that MUM should not take a sequence of actions (a diagnostic plan, or work-up) to gain evidence if another sequence would be as informative but less expensive. Viewing this as a traditional planning problem, we would project the outcomes of all sequences of evidence-gathering actions and select the sequence that provides us with maximum diagnosticity for minimum cost. Planning diagnoses is then a matter of searching this space.

In MUM we assume that the space of diagnostic plans or work-ups is too large to search exhaustively; that is, we will be unable to generate the most efficient work-up. We propose instead that work-ups are generated one action at a time or, equivalently, that the search for diagnostic plans proceeds incrementally, with each action being executed before the next is contemplated. MUM is essentially a reactive planner because its actions are determined largely by its environment (the state of the patient) and its preferences. The search is guided by heuristics that we call *preferences*. One preference is to ask cheap questions first; another is to ask diagnostic questions first. A more specific preference resolves the conflict that can arise between these two: If the patient is in danger, take the most diagnostic action regardless of cost, but if the patient is not in danger, take the cheaper action. In the worst case, we might require a specific preference for every situation that could arise, but in practice we can generate moderately efficient work-ups with relatively few preferences.

MUM has two components: an interpreter and an inference network with nodes representing evidence-gathering actions at the bottom, intermediate conclusions in the middle, and diseases at the top. MUM's interpreter selects and

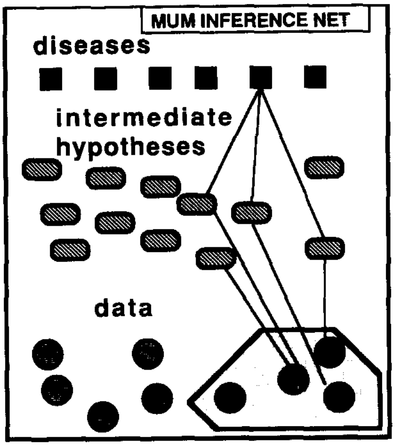


FIGURE 1. MUM Inference. Determine focus of attention then select an evidence-gathering Action.

executes an evidence-gathering action and propagates the data it acquires through the inference network; then, on the basis of its preferences and the new state of the network, it selects and executes another evidence-gathering action, and so on (see Figure 1).

The ASK and MU Systems

After building the MUM system, we abstracted its essential components and built an architecture called MU (Cohen et al. [16]). MU has been used to develop other systems like MUM that generate diagnostic work-ups via preferences. The central idea in MU is that decisions about actions are based on many *features* of a situation; for example, in medicine these include the cost, diagnosticity, risk, discomfort, and time required for a test; the number of supported disease hypotheses; the prior probabilities of these diseases, and whether any are serious; relationships between disease hypotheses such as causality or mutual exclusivity; relationships between actions and hypotheses, such as whether a test can discriminate two disease hypotheses; and contextual information such as whether the patient is feeble or robust.[6] MU allows knowledge engineers to rapidly define features and automatically updates them.

Features characterize the *states* of MU-based planners; for example, in one state we may have a *feeble* patient with *suggestive* evidence of a *dangerous* hypothesis, and we may have available a *completely diagnostic but risky* test, and another *moderately diagnostic* treatment that *protects* against heart attack.

[6] MUM did not incorporate all these features.

Given this characterization of the choice between actions, MU requires preferences to tell it which to do. To be useful, preferences should be general. That is, preferences should apply at many points in diagnostic plans, and over many plans. A Ph.D. student in our laboratory has developed a technique for acquiring and then generalizing expert preferences (Gruber [58]). The method, called ASK for Acquiring Strategic Knowledge, uses a small set of preferences to generate diagnostic work-ups that experts then criticize. The criticisms are used to specialize the original preferences and to add new, specific ones that are not in the original set. These are generalized if later criticisms suggest that their applicability is too narrow.

In sum, MU makes it easy to define and maintain the values of features, and the ASK system makes it easy to define preferences based on features. Features and their values make up the dynamic state of the planner, and states and preferences determine which problem-solving actions will be taken.

THE RELATIONSHIP BETWEEN PREFERENCES AND PLANS We rely on preferences to guide MUM's planning without any projection. Plans and preferences serve the same purpose, namely, to tell the problem solver what to do next. But plans require projection, while preferences do not. We claim that, in MUM at least, plans and preferences are related in such a way that sequences of actions based on preferences will appear to be good plans. The simplest such relationship between preferences and plans is that *preferences are local applications of the evaluation criteria for plans* (see Figure 2). For example,

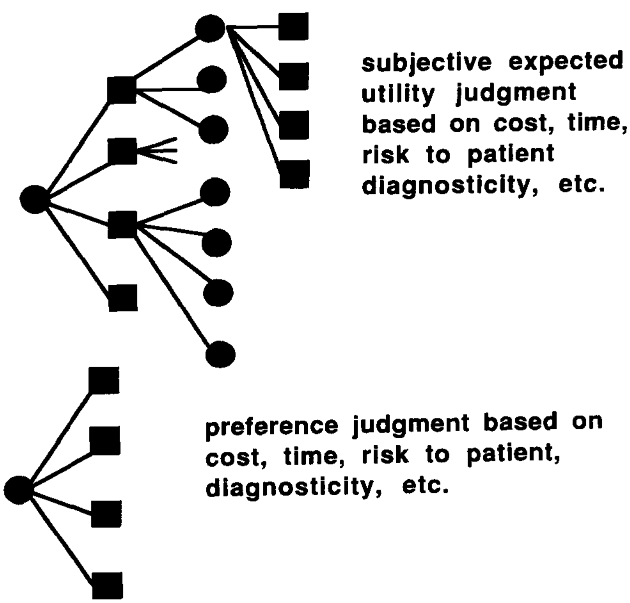


FIGURE 2. Plans versus preferences.

we noted earlier that if we prefer inexpensive plans, then we can generate plans without projection by locally selecting the cheapest applicable action. This will not always find the cheapest plan, but as a heuristic for guiding diagnostic work-ups, especially in combination with other preferences, it generates acceptable work-ups without the combinatorics of planning.

The relationship between plans and preferences is illustrated by the *work-up graph* for chest pain in Figure 3. This is an explicit contingency plan for the diagnosis of angina, generated by an expert internist and full of implicit preferences. Obviously, one can generate hundreds of other work-up graphs for chest pain by taking actions in different orders; for example, one might generate a plan that puts *angiogram* before *therapy*. But this and other syntactically possible work-ups violate expert preferences.[7] By following these preferences, one can generate a sequence of actions reactively, one at a time, that look as if they were planned in advance (as discussed earlier) and concur with Figure 3.

WHY WE NEED BOTH PREFERENCES AND PLANS Although MUM generates diagnostic plans from preferences alone, we believe that expert physicians plan, that is, project the outcomes of actions. They do not plan complete diagnostic work-ups, but they do some limited lookahead.[8] Here are two examples of lookahead in MUM's domain:

- *The cascade effect.* Some tests lead inexorably to others that you may want to avoid, and so should be avoided themselves. For example, you want to avoid a stress test if possible because unless the patient is absolutely cleared, you are obliged to go on to the next step, which is an angiogram. Now almost everyone has some degree of coronary artery blockage, and nobody knows how much is too much. So you start with a stress test that is not conclusive, and then you find some blockage, and then you are forced to do surgery, even though the patient may not have coronary artery disease.
- *Dependent tests.* If test 1 is diagnostic but costs a lot, and test 2 costs less but provides lower-quality evidence, then you may plan to do test 2 first and only do test 1 if test 2 comes back positive.

---

[7] For example, doing an angiogram before prescribing therapy violates at least four preferences: First, therapy provides evidence about angina that is more efficient, that is, a bit less diagnostic than an angiogram but much, much less expensive in terms of dollars, pain, and risk; second, therapy has few side effects; third, it provides evidence about the relevance of later tests, specifically evidence about whether the angiogram is necessary; fourth, therapy extends the time before the physician must take his or her "final action," which is surgery. Each of these is an argument, or preference, for therapy over an angiogram at a particular point in a work-up.

[8] The word lookahead suggests an analogy with game-tree search, in which preferences have the same role as static evaluation functions and projection is lookahead to some depth horizon. The analogy suggests that reactive planners are at one end of a continuum (they evaluate actions at a horizon of 1 or 0) and that planners can be more or less reactive depending on where they draw their depth horizon.

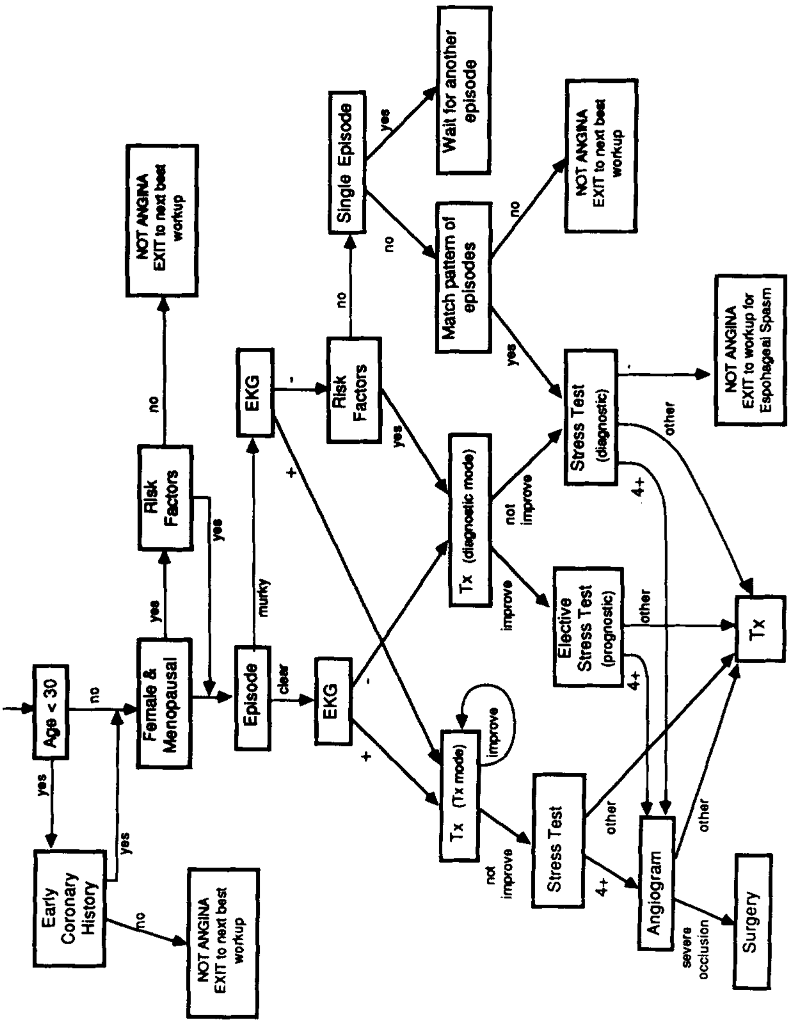


FIGURE 3. Workup for Angina (Note: Tx refers to nonsurgical treatment.).

In both these cases, the selection (or avoidance) of an action is based on projecting the outcomes of the actions.

These examples hint at two circumstances in which projection is desirable. In the cascade effect, projection detects pitfalls—situations where an attractive action leads to an unattractive one. A more familiar example of a pitfall is shown in Figure 4: When presented with a queen for the taking, my preferences say go ahead. The pitfall is that the queen is a sacrifice and checkmate follows. My preferences *should* say "Prefer piece exchanges that end with me ahead," but this requires projection. MUM was able to plan without projection because its search space of plans contained almost no pitfalls; however, some diagnostic plans are more *efficient* than others. Efficiency is the other reason for projection. If the order of actions affects the efficiency of plans, as it does whenever actions are not independent, then projection will contribute to efficiency.

In the following section, we describe a planning problem in which both pitfalls and efficiency are concerns. This problem involves real-time planning in highly uncertain environments. We are using it to study trade-offs between projection and reaction.

PLASTYC: Planning in Real-Time Dynamic Environments

We have built a large simulation of forest fires and the equipment commonly used to put them out. We are building a planner called PLASTYC that operates in this dynamic, real-time world. The planner's goal is to manage the fire—limit the loss of human life, limit the damage to forest and buildings, and limit the

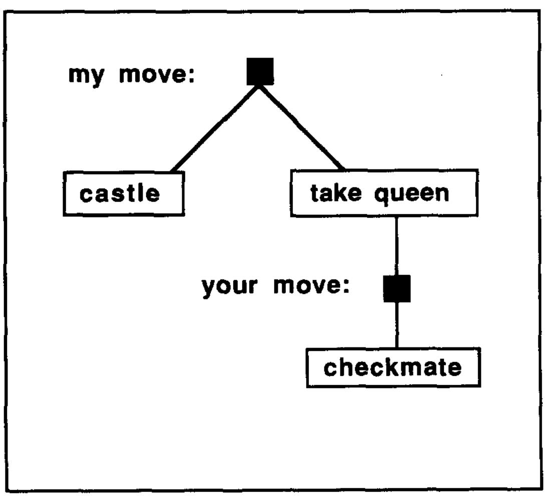


FIGURE 4. A decision in Chess.

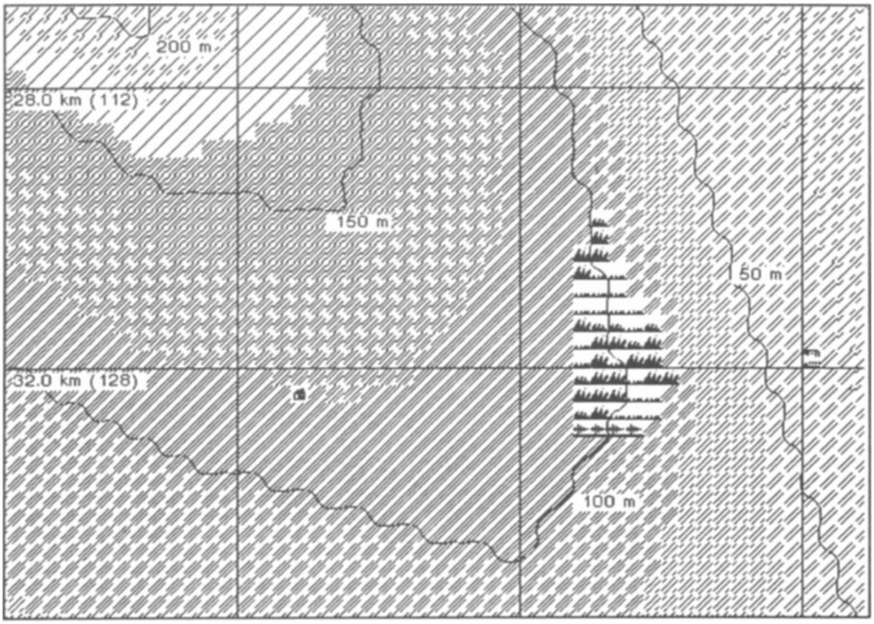
monetary costs of achieving these goals. This task closely approximates the problems faced by a forest fire manager, but, more important, it is representative of a class of problems that we believe require both traditional planning and the kind of reactive planning we used in MUM. In this section we describe the characteristics of this class of problems and sketch the planner we are building. This work is in progress, so the conclusions of this section are tentative.

Our simulation consists of a large geographical area (Explorer National Park) in which there is a considerable variety of topography and ground cover, as well as roads, lakes, and streams. These features affect how forest fires burn. Equally important features are wind speed and direction, both of which can change unpredictably, and the moisture content of the ground cover, which varies over time and geographically. To fight the fire, the simulation provides bulldozers, crews, transport vehicles, planes, and helicopters. These cut fire line, move firefighters, spray water, or dump retardant.

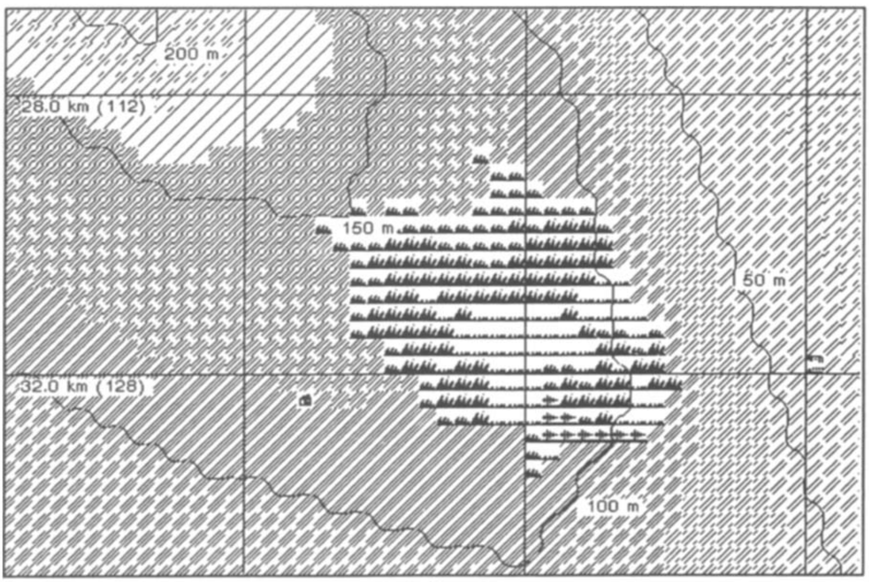
These fire-fighting agents can be directed either by an automatic planner or by a human player of what is essentially a complex, real-time video game. We have already gained considerable insight into (and respect for) the dynamics of this miniworld by playing against the simulation—often losing many lives and considerable real estate to a seemingly slow and containable fire. It is difficult for a planner, human or AI program, to do very well at the game (i.e., put out the fire with reasonable costs, no loss of life, etc.) because:

- The planner's knowledge of the fire is limited to what the agents in the field can "see." Crews and bulldozers can see only short distances; aircraft can see further. The planner rarely, if ever, has complete knowledge of the extent or location of the fire (see Figure 5).
- The behavior of the fire cannot be accurately predicted, because some factors that affect it (e.g., terrain, ground cover, and the moisture content of the ground cover) are known only approximately. Moreover, wind speed and direction can change unpredictably.
- The behavior of the fire-fighting agents cannot be accurately predicted. In particular, the time required to move to a location or perform some task depends on terrain and ground cover. Fire-fighting agents also have limited autonomy to run away from a fire, so the central planner cannot always be sure of their location.
- The simulation is real-time with respect to the fire. While fire-fighting agents move, cut line, and drop retardant, the fire keeps burning. Most important, any time the planner devotes to deliberation loses real estate to the fire.

The environment with which the planner interacts is independent, dynamic, and probabilistic. It is independent because the planner is not the only agent of change, dynamic because changes take place over time, and probabilistic because the magnitude and temporal extent of changes to the environment are unpredictable.



(a)



(b)

FIGURE 5. (*A*) The State of the Fire as It Can Be Seen by the Planner. (*B*) The State of the Fire as It Really Is.

To plan in this environment an agent needs to know when to project and when to plan reactively. Projection is desirable to avoid pitfalls and to increase the efficiency of plans. But projection itself takes time, uncertainty precludes avoiding *all* pitfalls, and efficient plans are useless if no time remains to execute them; so there will be situations when the timeliness of reaction will make the difference between success and failure.

We have described the desired behavior of PLASTYC but not how the program will get the knowledge it needs to behave that way. We have become adept at putting out fires in the forest fire simulation, and we will encode this knowledge in PLASTYC. But this is slow. PLASTYC itself should acquire and refine these skills through practice. The extent to which this can be accomplished remains to be seen; the project is in its early stages.

CONCLUSION

We argued in the Introduction that we should study reasoning under uncertainty in the context of autonomous action. In conclusion, we offer some methodological observations. In MUM, we wanted to study problem-solving strategies in medical diagnosis, but first we had to build an expert system, and then we had to acquire test cases from an expert. We wanted to generate thousands of problems for our planner and to evaluate the planner on objective criteria; neither was possible in internal medicine. We needed to challenge our planners with something like a game, but one that is played in a complex environment, in real time, under significant uncertainty. PLASTYC is designed to play autonomously against such a game—a simulation of forest fires. Not only does this simulated world provide an objective and efficient way to evaluate PLASTYC's abilities, but it will also present thousands of individual problems from which PLASTYC will begin to learn.

Our experience with PLASTYC suggests a general method for addressing problems in approximate reasoning. If you can build a simulator that presents agents with difficult problems in uncertain environments, and you can build agents that solve these problems autonomously, then you will have demonstrated unambiguous progress toward theories of action under uncertainty.

References

1. Tong, R. M., and Appelbaum, L. A., Problem structure and evidential reasoning, in *Third Workshop on Uncertainty in Artificial Intelligence,* American Association for Artificial Intelligence, 313–319, 1987.

2. Hanks, S., Temporal reasoning about uncertain worlds, in *Third Workshop on Uncertainty in Artificial Intelligence,* American Association for Artificial Intelligence, 114–122, 1987.

3. Cohen, P. R., Steps toward programs that manage uncertainty, in *Third Workshop on Uncertainty in Artificial Intelligence,* American Association for Artificial Intelligence, 372–379, 1987.

4. Wellman, M. P., Qualitative probabilistic networks for planning under uncertainty, in *Uncertainty in Artificial Intelligence,* (J. F. Lemmer, Ed.), 1987.

5. McDermott, D. V., DARPA-sponsored planning research: report and prospectus, YALE/CSD/RR 522, Yale University, Department of Computer Science, March 1987.

6. DARPA, *Proceedings of the Knowledge-Based Planning Workshop,* Austin, Texas, July 1987.

7. Georgeff, M. P., and Lansky, A. L., Eds., *The 1986 Workshop on Reasoning about Actions and Plans,* Timberline, Ore., 1987.

8. Lesser, V. R., Durfee, E. H., and Pavlin, J., Approximate processing in real-time problem solving, COINS Tech. Report 87-126, University of Massachusetts, Amherst, Mass., December 1987.

9. Barr, A., and Feigenbaum, E. A., Eds., *The Handbook of Artificial Intelligence,* Vol. 1, William Kaufmann, Inc., Los Altos, Cal., 1981.

10. Von Winterfeldt, D., and Edwards, W., *Decision Analysis and Behavioral Research,* Cambridge University Press, Cambridge, U.K., 1986.

11. Cohen, P. R., and Feigenbaum, E., *The Handbook of Artificial Intelligence,* Vol. 3, William F. Kaufmann, Los Altos, Cal., 1982.

12. Hammond, K., The use of remindings in planning, in *Proceedings of the Cognitive Science Society,* 1986.

13. Sycara, K., Planning for negotiation: a case-based approach, in *Proceedings of the DARPA Knowledge-Based Planning Workshop,* Austin, Texas, December 1987.

14. Gruber, T. R., and Cohen, P. R., Design for acquisition: principles of knowledge system design to facilitate knowledge acquisition, *Int. J. Man-Mach. Stud.* **26**(2), 143–159, 1987.

15. Cohen, P. R., Day, D. S., Delisio, J., Greenberg, M., Kjeldsen, R., Suthers, D., and Berman, P., Management of uncertainty in medicine, *Int. J. Approx. Reasoning* **1**(1), 103–116, 1987. Also *Proceedings of the IEEE Conference on Computers and Communications,* Phoenix, Ariz., February, 1987.

16. Cohen, P. R., Greenberg, M., and Delisio, J., MU: a development environment for prospective reasoning systems, in *Proceedings of the Sixth National Conference on Artificial Intelligence,* Seattle, Washington, 783–788, July 1987.

17. Cohen, P. R., *The Control of Reasoning Under Uncertainty: A Discussion of Some Programs,* Vol. 2, March 1987.

18. Simon, H. A., *The Sciences of the Artificial,* MIT Press, Cambridge, Mass., 1981.

19. Schneider, W., and Shiffrin, R. M., Controlled and automatic human information processing: I. detection, search and attention, *Psych. Rev.* 84, 1–66, 1977.

20. Schneider, W., Toward a model of attention and the development of automatic processing, in *Attention and Performance,* Vol. 11, (M. Posner and O. S. Marin, Eds.), Lawrence Erlbaum Associates, Hillsdale, N.J., 475–492, 1985.

21. Day, D. S., JANUS: an architecture for integrating automatic and controlled problem solving, in *Proceedings of the Ninth Annual Conference of the Cognitive Science Society,* Cognitive Science Society, Seattle, Wash., 655–662, 1987.

22. Day, D. S., Towards integrating automatic and controlled problem solving, in *Proceedings of the First International Conference on Neural Networks,* Institute of Electronic and Electrical Engineers, San Diego, Cal., 1987.

23. Michalski, R. S., Carbonell, J. G., and Mitchell, T. M., *Machine Learning: An Artificial Intelligence Approach,* Vol. II, Morgan Kaufmann, Palo Alto, Cal., 1986.

24. Sussman, G. J., *A Computer Model of Skill Acquisition,* Elsevier, New York, 1975.

25. Newell, A., and Simon, H. A., *Human Problem Solving,* Prentice-Hall, Englewood Cliffs, N.J., 1972.

26. Fikes, R., Hart, P., and Nilsson, N., Learning and executing generalized robot plans, *AI* **3**(4), 251–288, 1972.

27. Tate, A., INTERPLAN: a plan generation system that can deal with interactions between goals, Machine Intelligence Research Unit Memo MIP-1-109, University of Edinburgh, Edinburgh, December 1974.

28. Waldinger, R., Achieving several goals simultaneously, in *Machine Intelligence,* Vol. 8, E. W. Elcock and D. Michie, Eds.), Halstead/Wiley, New York, 1977.

29. Sacerdoti, E. D., Planning in a hierarchy of abstraction spaces, *AI* **5**(2), 115–135, 1974.

30. Sacerdoti, E. D., *A Structure for Plans and Behavior,* Elsevier, New York, 1975.

31. Tate, A., Generating project networks, in *Proceedings of the Fifth International Joint Conference on Artificial Intelligence,* Tiblisi, Georgia, USSR, 888–893, 1977.

32. Wilkins, D. E., Recovering from execution errors in SIPE, Tech. Report, 1985.

33. Broverman, C. A., and Croft, W. B., Reasoning about exceptions during plan execution, in *Proceedings of the Sixth National Conference on Artificial Intelligence,* Seattle, Washington, 190–195, 1987.

34. McDermott, D., A temporal logic for reasoning about processes and plans, *Cognitive Sci.* **6**, 101–155, 1982.

35. Allen, J. F., Towards a general theory of action and time. *AI* **23**(2), 123–154, 1984.

36. Dean, T., Large-scale temporal data bases for planning in complex domains, in *Proceedings of the Tenth International Joint Conference on Artificial Intelligence,* Milan, Italy, 1987.

37. Vere, S., Planning in time: windows and durations for activities and goals, Tech. Report, Jet Propulsion Laboratory, Pasadena, Calif., 1981.

38. Durfee, E. H., A unified approach to dynamic coordination: planning actions and interactions in a distributed problem solving network, Ph.D. dissertation, University of Massachusetts, Amherst, Mass., September 1987.

39. Hayes-Roth, B., Garvey, A., Johnson, M. V., and Hewett, M., A layered environment for reasoning about action, Tech. Report KSL 86-38, Computer Science Department, Stanford University, April 1986.

40. Garvey, A., Cornelius, C., and Hayes-Roth, B., Computational costs vs. benefits of control reasoning, in *Proceedings of AAAI-87: Sixth National Conference on Artificial Intelligence,* American Association for Artificial Intelligence, 1987.

41. Pardee, W. J., and Hayes-Roth, B., Intelligent real time control of material processing, Tech. Report, Rockwell International, Science Center, Palo Alto Laboratory, February 1987.

42. Korf, R., Real-time heuristic search: first results, in *Proceedings of the Sixth National Conference on Artificial Intelligence,* Seattle, Washington, 133–138, 1987.

43. Dacus, J. R., Knowledge-based planning in advanced tactical aircraft, in *Proceedings of the DARPA Knowledge-Based Planning Workshop,* Austin, Texas, December 1987.

44. Luhrs, R. A., and Nowicki, A. R., Real-time dynamic planning for autonomous vehicles, in *Proceedings of the DARPA Knowledge-Based Planning Workshop,* Austin, Texas, December 1987.

45. Herman, M. and Albus, J. S., Real-time hierarchical planning for multiple mobile robots, in *Proceedings of the DARPA Knowledge-Based Planning Workshop,* Austin, Texas, December 1987.

46. Daily, M., Harris, J., Deirsey, D., Olin, K., Payton, D., Reiser, K., Rosenblatt, J., Tseng, D., and Wong, V., Autonomous cross-country navigation with the ALV, in *Proceedings of the DARPA Knowledge-Based Planning Workshop,* Austin, Texas, December 1987.

47. Firby, R. J., and Hanks, S., A simulator for mobile robot planning, in *Proceedings of the DARPA Knowledge-Based Planning Workshop,* Austin, Texas, December 1987.

48. Hendler, J. A., and Sanborn, J. C., A model of reaction for planning in dynamic environments, in *Proceedings of the DARPA Knowledge-Based Planning Workshop,* Austin, Texas, December 1987.

49. Chapman, D., and Agre, P. E., Abstract reasoning as emergent from concrete activity, in *Reasoning About Actions and Plans, Proceedings of the 1986 Workshop at Timberline, Oregon* (M. P. Georgeff and A. L. Lansky, Eds.), 411–424, 1987.

50. Agre, P. E., and Chapman, D., Pengi: An implementation of a theory of activity, in

Proceedings of the Sixth National Conference on Artificial Intelligence, American Association for Artificial Intelligence, Seattle, Wash., 268–272, 1987.

51. Brooks, R. A., A robust layered control system for a mobile robot, A.I. Memo 864, Massachusetts Institute of Technology, 1985.

52. Firby, R. J., An investigation into reactive planning in complex domains, in *Proceedings of the Sixth National Conference on Artificial Intelligence,* Morgan Kaufmann, Seattle, Washington, 202–206, 1987.

53. Dean, T., Planning, execution and control, in *Proceedings of the DARPA Knowledge-Based Planning Workshop,* Austin, Texas, December 1987.

54. Barto, A. G., Sutton, R. S., and Anderson, C. W., Neuronlike adaptive elements that can solve difficult learning control problems, *IEEE Trans. Syst., Man Cybern.* SMC-13(5), September 1983.

55. Sutton, R. S., and Barto, A. G., Toward a modern theory of adaptive networks: expectation and prediction, *Psych. Rev.* **88**(2), 135–170, 1981.

56. Jordan, M. I., Attractor dynamics and parallelism in a connectionist sequential machine, in *Proceedings of the Cognitive Science Society,* 1986.

57. Rumelhart, D. E., and Norman, D. A., Simulating a skilled typist: a study of skilled cognitive-motor performance, *Cogn. Sci.* **6**, 1–36, 1982.

58. Gruber, T. R., Acquiring strategic knowledge from experts, in *Proceedings of the Second AAAI Knowledge Acquisition for Knowledge-Based Systems Workshop,* Banff, Canada, October 1987. To appear in *Int. J. Man-Mach. Stud.*

59. Laird, J. E., Rosenbloom, P. S., and Newell, A., Chunking in SOAR: the anatomy of a general learning mechanism, *Mach. Learning* **1**, 11–46, 1986.

60. Mitchell, T. M., Utgoff, P. E., and Banerji, R. B., Learning by experimentation: acquiring and refining problem solving heuristics, in *Machine Learning: An Artificial Intelligence Approach,* Michaliki, R. S., Carbonell, J. G., and Mitchell, T. M., Eds., Tioga: Palo Alto, Cal., 1983.

61. Anderson, J. R., *The Architecture of Cognition,* Harvard University Press, Cambridge, Mass., 1983.

62. Hayes-Roth, B., Dynamic control planning in adaptive intelligent systems, in *Proceedings of the DARPA Knowledge-Based Planning Workshop,* Austin, Texas, December 1987.

63. Swartout, W., Summary report on DARPA Santa Cruz workshop on planning, in *Proceedings of the DARPA Knowledge-Based Planning Workshop,* Austin, Texas, December 1987.

64. Wellman, M. P., and Heckerman, D. E., The role of calculi in uncertain reasoning, in *Third Workshop on Uncertainty in Artificial Intelligence,* American Association for Artificial Intelligence, 321–331, 1987.