(1)

UMIACS-TR-90-21
CS-TR-2408

February 1990

AI Planning: Systems and Techniques

James Hendler, Austin Tate, and Mark Drummond†*
Institute for Advanced Computer Studies and
Department of Computer Science
University of Maryland
College Park, MD 20742

COMPUTER SCIENCE
TECHNICAL REPORT SERIES

UNIVERSITY OF MARYLAND
COLLEGE PARK, MARYLAND
20742

1

UMIACS–TR–90–21 February 1990
CS–TR–2408

AI Planning: Systems and Techniques

James Hendler, Austin Tate, and Mark Drummond†*
Institute for Advanced Computer Studies and
Department of Computer Science
University of Maryland
College Park, MD 20742

(article in AI Magazine)

ABSTRACT

This article, to appear in *AI Magazine* (Summer, 1990), reviews work done in the area of AI Planning Systems. The goal of this article is to familiarize the reader with some of the important problems which have arisen in the design of planning systems, and some of the many solutions which have been developed in the over thirty years of planning research. In this paper we provide a broad coverage of the major ideas in the field of AI planning and show the direction in which some current research is going. We define some of the terms commonly used in the planning literature, describe some of the basic issues arising out of the design of planning systems, and survey results in the area. As such a task is virtually never-ending, and any finite document must per force be incomplete, we also provide references to connect each idea to the appropriate literature and to allow readers access to the papers most relevant to their own research or applications work.

(12P)

*Artificial Intelligence Applications Institute, University of Edinburgh.

†Sterling Software, AI Research Branch, NASA Ames Research Center.

AI Planning: Systems and Techniques

James Hendler

Computer Science Department
University of Maryland

Austin Tate

Artificial Intelligence Applications Institute
University of Edinburgh

Mark Drummond

Sterling Software
AI Research Branch
NASA Ames Research Center

1. Introduction

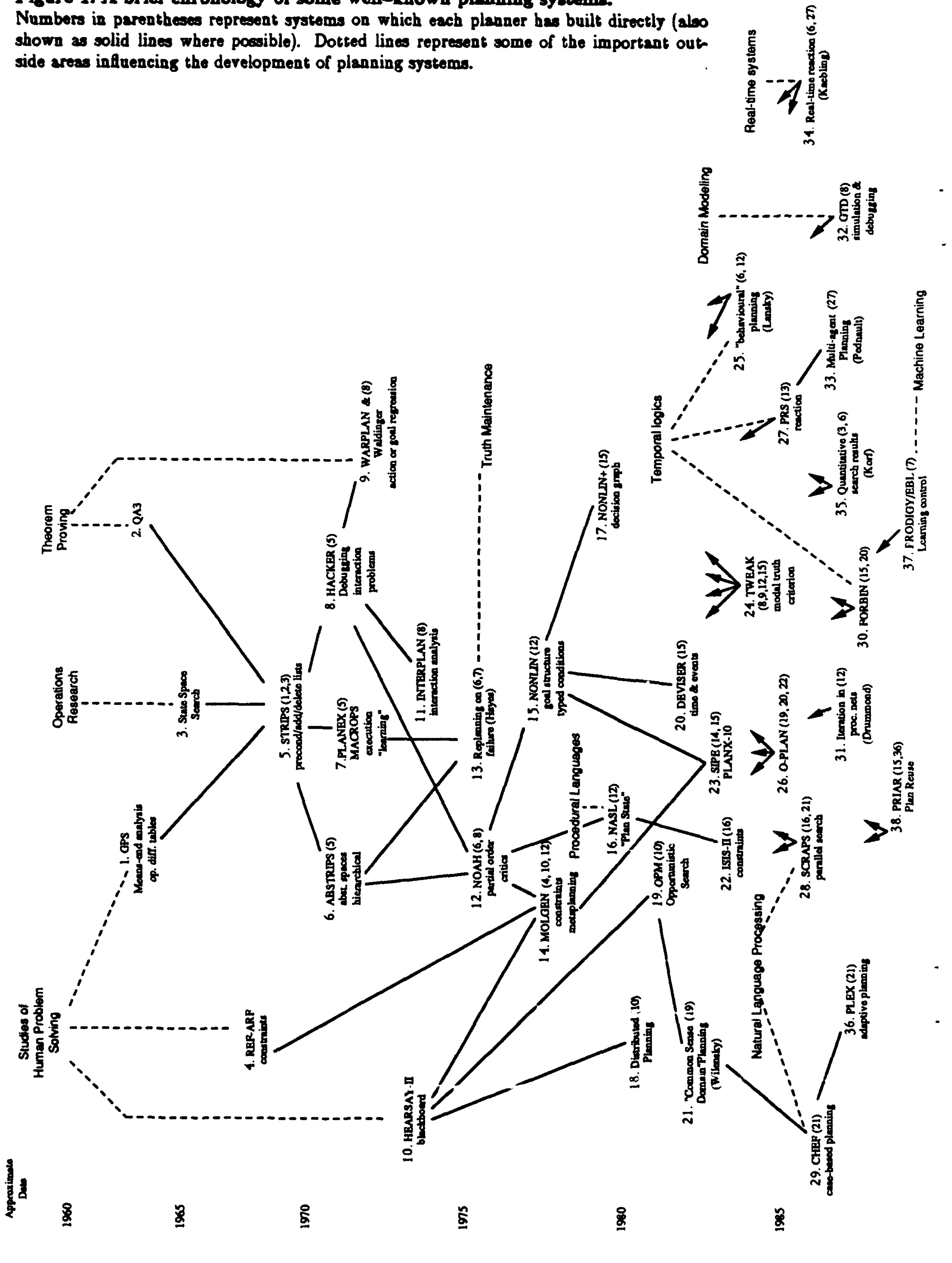
A long standing problem in the field of automated reasoning is that of designing systems which can describe a set of actions (or plan) which can be expected to allow the system to reach a desired goal. Ideally, the set of actions so produced is then passed on to a robot, a manufacturing system, or some other form of effector, which can follow the plan and produce the desired result. The design of such "planners" has been with AI since its earliest days, and a large number of techniques have been introduced in progressively more ambitious systems over a long period. In addition, planning research has introduced many problems to the field of AI. Some examples include:

- Representing and reasoning about time, causality and intentions
- Physical or other constraints on suitable solutions
- Uncertainty in the execution of plans
- Sensation and perception of the "real world" and holding beliefs about it
- Multiple agents who may cooperate or interfere

Planning problems, like most AI topics, have been attacked in two major ways: approaches which try to understand and solve the general problem without use of domain specific knowledge, and approaches which use domain heuristics directly. In planning, these are often referred to as ***domain-dependent*** approaches (which use domain-specific heuristics to control the planner's operation), and ***domain-independent*** approaches (in which the planning knowledge representation and algorithms are expected to work for a reasonably large variety of application domains). The issues involved in the design of domain-dependent planners are those generally found in applied approaches to AI — the need to justify solutions, the difficulty of knowledge acquisition, and the fact that the design principles may not map well from one application domain to another.

An early version of this article appeared in the *Knowledge Engineering Review* Volume 1 Number 2 pages 4-17, June 1985. Part of that review along with a tutorial introduction to the field appeared in the planning chapter of *Knowledge Engineering* Volume 1 published by McGraw-Hill (editor H.Adeli) in 1990. A version of this paper stressing technical definitions appears in the book ***Readings in Planning***, (Allen, Hendler, and Tate eds., 1990).

Figure 1: A brief chronology of some well-known planning systems. Numbers in parentheses represent systems on which each planner has built directly (also shown as solid lines where possible). Dotted lines represent some of the important outside areas influencing the development of planning systems.



Approximate Date
1960
1965
1970
1975
1980
1985
Studies of Human Problem Solving
Operations Research
Theorem Proving
1. GPS Means-end analysis op. diff. tables
2. QA3
3. State Space Search
4. REF-ARF constraints
5. STRIPS (1,2,3) precond/add/delete lists
6. ABSTRIPS (5) abs. spaces hierarchical
7. PLANEX (5) MACROPS execution "learning"
8. HACKER (5) Debugging interaction problems
9. WARPLAN & (8) Waldinger action or goal regression
10. HEARSAY-II blackboard
11. INTERPLAN (8) interaction analysis
12. NOAH (6, 8) partial order critics
13. Replanning on (6,7) failure (Hayes)
Truth Maintenance
14. MOLGEN (4, 10, 12) constraints metaplanning
Procedural Languages
15. NONLIN (12) goal structure typed conditions
16. NASL (12) "Plan State"
17. NONLIN+ (15) decision graph
18. Distributed (10) Planning
19. OPM (10) Opportunistic Search
20. DEVISER (15) time & events
21. "Common Sense (19) Domain" Planning (Wilensky)
22. ISIS-II (16) constraints
23. SIPE (14, 15) PLANX-10
24. TWEAK (8,9,12,15) modal truth criterion
25. "behavioural" (6, 12) planning (Lansky)
26. O-PLAN (19, 20, 22)
27. PRS (13) reaction
28. SCRAPS (16, 21) parallel search
29. CHEF (21) case-based planning
30. FORBIN (15, 20)
31. Iteration in (12) proc. nets (Drummond)
32. GTD (8) simulation & debugging
33. Multi-agent (27) Planning (Pednault)
34. Real-time reaction (6, 27) (Kaebling)
35. Quantitative (3, 6) search results (Korf)
36. PLEX (21) adaptive planning
37. FRODIGY/EBL (7) Learning control
38. PRIAR (15,36) Plan Reuse
Temporal logics
Domain Modeling
Real-time systems
Machine Learning
Natural Language Processing

Planner	Domain
STRIPS (Fikes & Nilsson, 1972a)	Simple Robot Control
HACKER (Sussman, 1973)	Simple Program Generation
NOAH (Sacerdoti, 1977)	Mechanical Engineers apprentice supervision
NONLIN (Tate, 1977)	Electricity Turbine Overhaul
NASL (McDermott, 1978)	Electronic Circuit Design
OPM (Hayes-Roth & Hayes-Roth, 1979)	Journey Planning
ISIS-II (Fox et. al., 1981)	Job Shop Scheduling (Turbine Production)
MOLGEN (Stefik, 1981a)	Experiment Planning in Molecular Genetics
SIPE (Wilkins, 1983)	Aircraft Carrier Mission Planning
NONLIN+ (Tate & Whiter, 1984)	Naval Logistics
DEVISER (Vere, 1983)	Voyager Spacecraft Mission Sequencing
FORBIN (Miller et. al., 1985)	Factory Control

Table I: Numerous planning systems have attempted to integrate available planning techniques and apply them to application domains.

Figure 1 about here.

Work in domain-independent planning has formed the bulk of the AI research in the area of planning. The long history of these efforts (see figure 1) has led to the discovery of many recurring problems, as well as to certain standard solutions. In addition, there have been a number of attempts to combine the planning techniques available at a given time into prototypes able to cope with increasingly more realistic application domains. (Table 1 shows an example of some of these efforts and the domains to which they were applied.)

Table 1 about here.

The goal of this article is to familiarize the reader with some of the important problems which have arisen in the design of planning systems, and some of the many solutions which have been developed in the over thirty years of planning research. In this paper we will try to provide a broad coverage of the major ideas in the field of AI planning and attempt to show the direction in which current research is going. We will define some of the terms commonly used in the planning literature, describe some of the basic issues arising out of the design of planning systems, and survey results in the area. (Due to the recurrence of many themes throughout the planning literature, we will survey the field on the basis of areas of interest rather than in terms of its chronological development.) Such a task is virtually never-ending, and any finite document must per force be incomplete. Thus, in addition to our discussion of the issues we have provided references to connect each idea to the appropriate literature and to allow readers access to the papers most relevant to their own research or applications work.

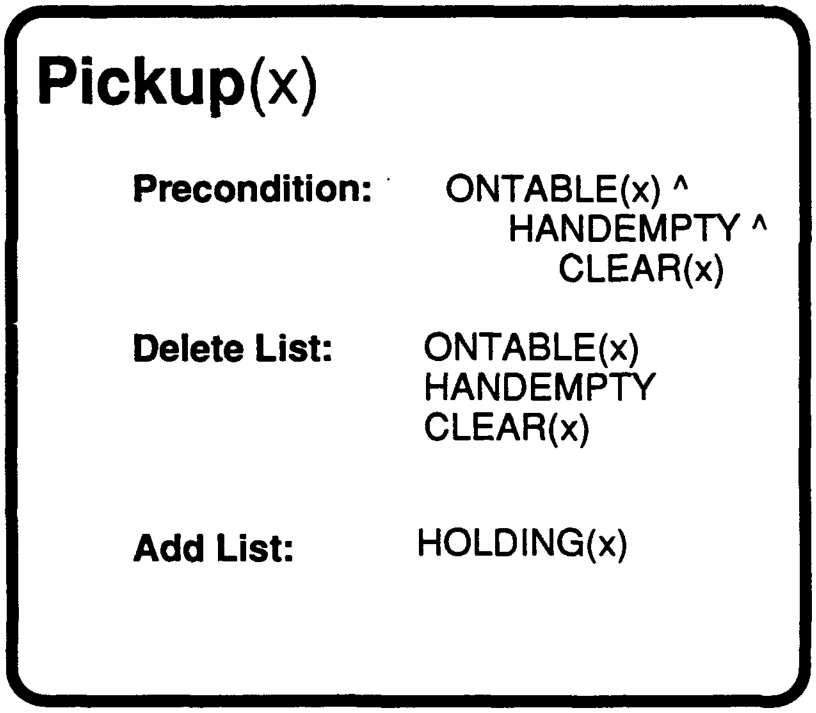
2. Planning Terminology

An AI planning system is charged with generating a *plan* which is one possible *solution* to a specified *problem*. The plan generated will be composed out of *operator schemata*, provided to the system for each domain of application. This section briefly considers the meaning of each of these terms and how they relate to one another.

A problem is characterized by an initial state and goal state description. The initial state description tells the planning system the way the world is "right now". The goal state description tells the planning system the way we want the world to look when the plan has been executed. The world in which planning takes place is often called the *application domain*. We will sometimes refer to the goal state description as simply "the goal." In many systems, a goal may be transformed into a set of other, usually simpler, goals called "sub-goals."

Operator schemata characterize *actions*. (The terms *action* and *event* are often used interchangeably in the AI planning literature, and certainly will be here.) *Schemata* primarily describe actions in terms of their preconditions and effects. Plans are built out of these operator schemata. Each operator schemata characterizes a *class* of possible actions, by containing a set of variables which can be replaced by constants to derive operator *instances* that describe specific, individual, actions. When the distinction doesn't matter, we'll use the term *operator* to stand for both operator schemata and operator instances. An action which the planner considers to be directly executable is referred to as a *primitive action*, or simply as a *primitive*.

Figure 2 about here.



Pickup(x)

Precondition: ONTABLE(x) ^ HANDEMPTY ^ CLEAR(x)

Delete List: ONTABLE(x)
HANDEMPTY
CLEAR(x)

Add List: HOLDING(x)

Figure 2: A typical STRIPS operator.
The operator *Pick-up* contains a precondition formula, add and delete lists. Thus, pick-up can be used to lift an object which must be on the table, the hand must be empty, and the object must be clear (from the preconditions). After pickup, the object is not on the table and is not clear, and the hand is no longer empty (from the delete-list), also the hand is known to be holding the object (from the add-list).

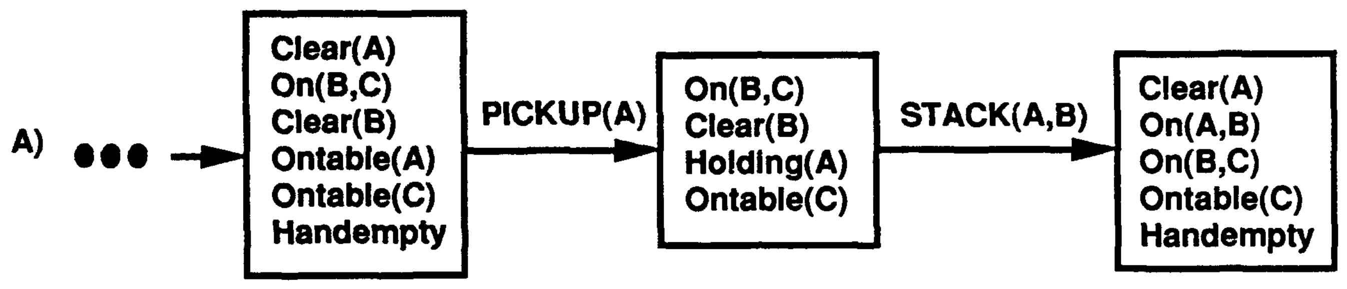
A commonly used terminology throughout the AI planning literature is that of *STRIPS operators* (Fikes, et.al., 1972a; 1972b). These operators, first used in the early planning program STRIPS, describe an action by means of three parts: a *precondition* formula, an *add-list* and a *delete-list* (see Figure 2). An operator's precondition formula (or simply, the operator's preconditions) give facts that must hold before the operator can be applied. The add-list and delete-list are used in concert to simulate action occurrence. If an operator's preconditions hold in a state then the operator can be applied. Applying an operator means acting on its add-list and delete-list to produce a new state. The new state is produced by first deleting all formulas given in the delete-list and then adding all formulas in the add-list. Although newer planning systems (discussed later) do depart from this approach, the terminology of STRIPS operators is fairly standard and we will use it often in this article.

A plan is an organized collection of operators. A plan is said to be a *solution* to a given problem if the plan is applicable in the problem's initial state, and if after plan execution, the goal is true. What does it mean for a plan to be "applicable"? Assume that there is some operator in the plan that must be executed first. Then the plan is *applicable* if all the preconditions for the execution of this first operator hold in the initial state. Repeated analysis can determine whether or not all operators can be applied in the order specified by the plan. This analysis is referred to as *temporal projection*. The first state considered in the projection is the problem's initial state. Repeated operator applications produce intermediate state descriptions. If the goal is true at the end of this projection then the plan is a solution to the specified problem.

So, the inputs to a typical AI planning system are a set of operator schemata and a problem that is characterized by an initial state description and goal. The output from the planner is a plan which under projection satisfies the goal. The process connecting the input and output is known by various names. Common names are plan *generation*, plan *synthesis* and plan *construction*. A planner is called *domain independent* in the sense that the plan representation language and plan generation (or synthesis, or construction) algorithms are expected to work for a reasonably large variety of application domains. While this view of planning is slightly restrictive, it will suffice for the purposes of this overview.

A planner is organized so that it defines a search space and then seeks a point in that search space which is defined as a solution. Planners differ as to how they define their search space. Some (most of the pre-1975 planners) define points in the search space as "states" of the application's world at various times. This "world state" can be traversed by applying any applicable operator to some chosen state leading to a different point in the search space. A problem solution can be defined as a sequence of operators that can traverse the search space from some initial state to some state defined as satisfying the goal. Thus, a state-space plan contains descriptions of states of the world the plan is to be executed in, and move operators which correspond to the actions to be carried out to execute the plan. A state-space solution is trivial to recognize: it is quite simply a path from some initial state to a state satisfying the goal specification (see Figure 3a).

Other (most of the post-1975) planners define points in the search space as partially elaborated plans. One point in this space is changed into another using any applicable planning transformation such as the expansion of an action to a greater level of detail, the inclusion of an additional ordering constraint between actions to resolve some interaction between effects of unordered actions, etc.. Given some initial (skeleton) plan defining a point in this "partial plan search space", a sequence of plan transformations must be applied which lead to a fully detailed plan that satisfies the goal (see Figure 3b). A plan is considered to be complete when all of the goals can be realized as a set of primitive actions (totally- or partially-ordered depending on the planner).



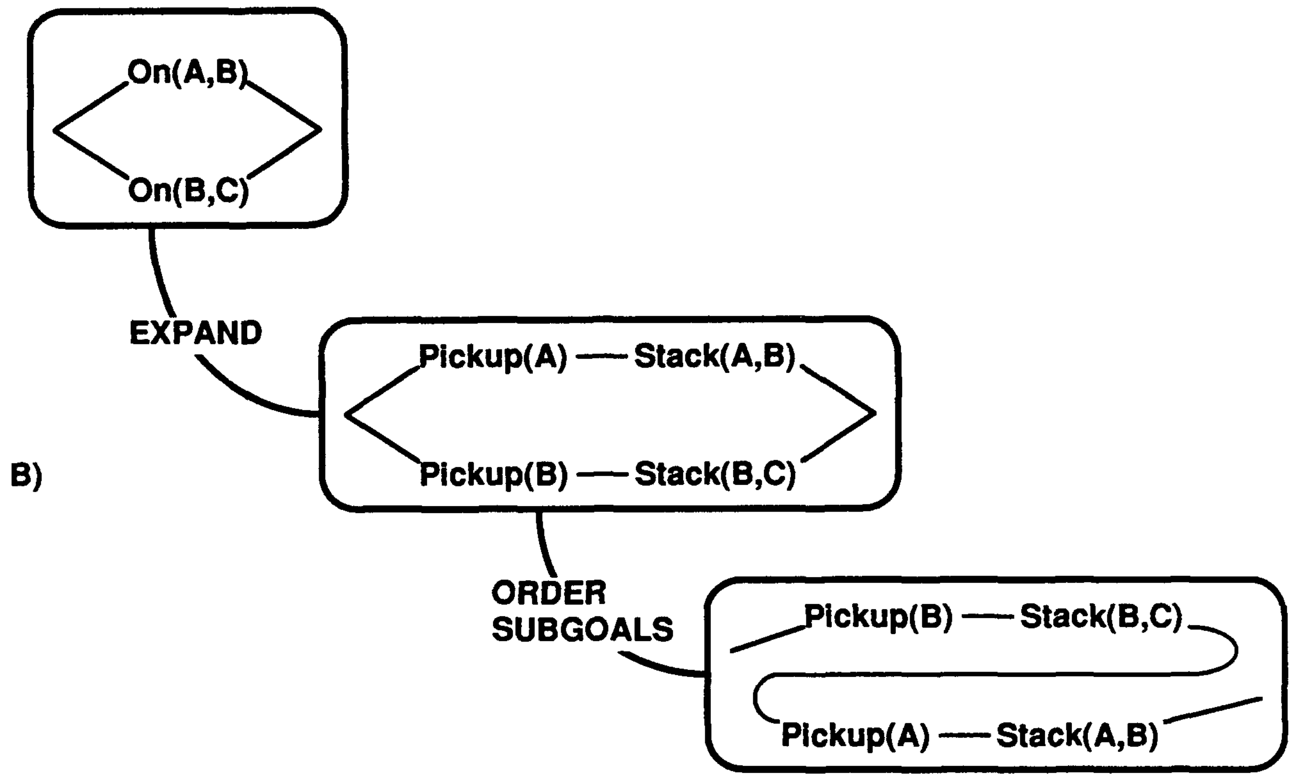


Figure 3: Partial solutions to the blocks world problem "(ON A B) & (ON B C)"
Figure 3a (based on STRIPS) shows a typical state-space and a sequence of operators, representing a partial solution. Figure 3b (based on NOAH) shows the plan as a partial ordering of actions representing a partial solution.

Figure 3(a,b) about here.

Systems which search through the space of partial plans have typically represented plans as an "action-ordering" in which the actions, described by operators, are strung together with temporal ordering relations. The most famous example of such plans are Sacerdoti's (1975) *Procedural Nets* (see section 3). Action-ordering plans describe the relationships among actions directly, rather than through states and predicates contained within states.

In an action-ordering representation a plan need not completely specify the conditions that each action affects. The plan can simply order two actions without specifying intermediate states. In contrast, state-based plan structures typically require complete specification of these intermediate states. This is particularly difficult for use in describing complicated causal and temporal relationships *between* actions. Thus, many complex domains are quite difficult to encode using a state-based approach (cf. Lansky, 1987), and action-ordering approaches have become the more generally used technique.

3. Planning and Search

As can be seen from the discussion above, planning is essentially a search problem. The program must traverse a potentially large search space and find a plan which is applicable in the initial state and produces a solution to the goal when run. This search can be quite difficult as the search space may contain many interactions between different states or partial plans. These interactions lead to a surprising amount of complexity — for example, establishing the existence of a precondition in a partially ordered plan may require exponential computation (Chapman, 1987) and the problem of finding an optimal plan in even a simple Blocks World domain has recently been shown to be NP-hard (Gupta & Nau, 1990).† The problems of organizing the heuristic search, choosing what to do in cases of failure, and generally finding ways of making more informed choices have therefore been among the most discussed in the planning literature. (A good quantitative discussion of planning viewed as heuristic search can be found in Korf (1987).)

Early approaches to planning sought to apply legal "moves" to some initial state to search for a state that satisfied the given goals. There was a great deal of overlap with game playing work. Heuristic evaluation functions were employed to rate the various intermediate states produced to estimate their "closeness" to a goal state (e.g. in A* (Hart et al, 1968) and the Graph Traverser (Doran & Michie, 1966)). This approach, however, was found wanting due to the difficulty in designing working heuristics and the usually exponential growth of the search space .

To reduce the number of intermediate states considered, Newell and Simon (1963) introduced "means-ends anlysis", a heuristic which involves considering only those activities that could satisfy some outstanding goal. Operators which when run, would cause the current state to more closely resemble the goal state, were preferred. This technique was used as the basis of the search in many of the early planners which used state-space-based plans.

With the introduction of *Procedural Nets* in NOAH (Sacerdoti, 1975), the search problem changed. In this system and its descendants, the search space consists not of a set of 'world states,' but rather a space of partial plans. For any non-primitive action in the current network the planner may consider any known method of "reducing" that action to a set of other actions and/or primitives. Planning in such a system consists of choosing appropriate reductions from

† Planning involving conditional effects has been shown to be undecidable(Chapman, 1987) — that is, a plan generated in such domains cannot be guaranteed to succeed without some type of execution time additions. This issue is discussed in Section 6.

among the sets of possibilities and ordering actions so as to eliminate harmful interactions.

One important technique introduced for searching partial plans was that of least commitment plan representations[†]. Such representations are used to allow a set of plans to be represented in a single state of the search. Examples of such representations include the use of a partially-ordered plan to represent a number of possible action orderings prior to a commitment becoming necessary (e.g. in NOAH, Sacerdoti 1975), or the posting of constraints on objects referred to in the plan rather than making an arbitrary selection (e.g. in MOLGEN, Stefik 1981a).

To search through the space of partially-ordered plans many solutions have been offered. Some systems do not search through the possible alternatives at all. Instead, selections are made on the basis of locally available information and a commitment is made to a particular solution path. (Of course, this means that some problems cannot be solved.) This technique, has been most successful where strong domain heuristics can be used for making the choices. Where such heuristics are not available, a more general solution is to use backtracking to allow backing-up to occur when the goal cannot be reached based on some earlier choice. The planning system simply saves the state of the solution at each point at which there are alternative ways to proceed and keeps a record of the alternative choices. The first is chosen and search continues. If there is any failure, the saved state at the last choice point is restored and the next alternative taken (if there are none, "backtracking" continues over previous decisions). Simple stack based implementation techniques can be used for this process (for example, as done in Prolog).

The NONLIN program (Tate, 1977) introduced a variant of depth-first backtracking for use in planning. Since there is often good local information available to indicate the preferred solution path, it is often appropriate to try the best choice indicated by local heuristic information before considering the many alternatives that may be available should the local choice prove faulty. Taken to the extreme, depth-first search gives something of the flavor of such a search strategy. However, gradual wandering from a valid solution path could entail backtracking through many levels when a failure is detected. An alternative is to focus on the choice currently being made and try to select one of the local choices which seems most promising. This continues while all is going well (perhaps with some cut-off points to take a long, hard look at how well things are going). However, if a failure occurs, NONLIN considers the *entire* set of alternatives which have been generated (and ranked by a heuristic evaluator).

This basic technique has been further refined to provide the most widely used approach to controlling search in planning. Any backtracking system based on saved states and resumption points (whether depth-first or heuristically controlled) can waste much valuable search effort — there may be several unrelated parts to a solution. If it so happens that backtracking on one part has to go back beyond points at which work was done on an unrelated part, all effort on the unrelated part will be lost. Many planning systems avoid this problem by using a variant on the backtracking methods used in NONLIN. These systems do not keep saved states of the solution at choice points. Instead, they record the dependencies between decisions, the assumptions on which they are based, and the alternatives from which a selection can be made. They then use methods for undoing a failure by propagating and undoing all of the dependent parts of the solution. This leaves unrelated parts intact irrespective of whether they were worked on after some undone part of the solution. Examples of work using this technique include Hayes (1975), Stallman and Sussman (1977), NONLIN+Decision Graph (Daniel, 1983) and MOLGEN (Stefik, 1981a,b) to some extent.

One alternative that has been suggested to the use of backtracking based search is that of "opportunistic" planning systems. These systems do not take a fixed (goal-driven or data-

[†] Some people use the term "least commitment" to refer only to the ordering of plan steps in a partial-order planner. We use the term in the broader sense, refering to any aspect of a planner which only commits to a particular choice when forced by some constraints.

driven) approach to solving a problem. Instead, a current "focus" for the search is identified on the basis of the most constrained operation that can be performed. This may be suggested by comparison of the current goals with the initial world model state, by consideration of the number of likely outcomes of making a selection, by the degree to which goals are instantiated, etc. Any problem solving component may summarize its requirements for the solution as constraints on possible solutions or restrictions of the values of variables representing objects being manipulated. They can then suspend their operation until further information becomes available on which a more definite choice can be made (e.g. in MOLGEN, Stefik 1981a).

There are a number of planning systems which have an operator-like representation of the different types of plan transformations available to the planner. A separate search is made to decide which of these is best applied at any point; this happens before decisions are taken about the details of the particular application plan being produced (e.g., MOLGEN, Stefik 1981b; Wilensky 1981a; OPM, Hayes-Roth & Hayes-Roth 1979; PRS, Georgeff & Lansky 1987). This technique, primarily used in conjunction with opportunistic planning, is often refereed to as "metaplanning" as it requires the planner to reason not only about the goal, but also about the various techniques available for generating the plan.

Besides the basic method of reducing the search space by selection of relevant operators, many other methods have also been employed in planners. Some examples include Giving goals levels of priority and considering the highest priority goals first (cf. ABSTRIPS, Sacerdoti 1973; LAWALY, Siklossy & Dreussi 1975) or rejecting states or plans that are known to be impossible or in violation of some rule (cf. WARPLAN, Warren, 1974). (This latter has been further refined to include the use of domain constraints (Allen and Koomen, 1983), and temporal coherence (Drummond & Currie 1988, 1989) to provide heuristics for rejecting possible actions). A variant of this approach, using a domain model to simulate the results of planning operators was used in Simmons & Davis' (1987) Generate-Test-Debug planner. DEVISER (Vere, 1983), SIPE (Wilkins, 1983), NONLIN+ (Tate & Whiter, 1984) and O-Plan (Currie & Tate, 1985) use checks on resource usage levels, time constraints on actions, and other resource bounds to eliminate some possibilities. A more recent approach has involved the use of parallelism to briefly examine many potential choices concurrently. Heuristics are used which allow the effects of various interactions detected by the parallel search to suggest possible choices or to rule out potential plans (cf. SCRAPS, Hendler 1987). Another approach is the use of natural occuring "locality" which may occur in a domain to reduce the search burden by partitioning the planning search space into smaller, localized search spaces (cf. GEMPLAN, Lanksy 1988).

4. Conjunctive Goal Planning

In addition to the problem of controlling the search, the order in which several simultaneous goals are tackled can have a marked effect on the efficiency of the search process. Some early planners, for example, could loop around on the same goals repeatedly or get redundant solution (in the sense that the final plan contained steps which could be removed without negating the plan's reaching of the goal) when the goals were attempted in the wrong order. The solving of such "conjunctive goal" plans has been the basis of much of the modern planning research. Two somewhat orthogonal approaches have been taken to dealing with this problem: ordering the various goals by levels of importance, and analyzing and avoiding the interactions caused by interactions between conjunctive goals.

The use of levels to partially overcome this problem (it is **not** a complete solution) was introduced in ABSTRIPS (Sacerdoti, 1973) and LAWALY (Siklossy & Dreussi, 1975). These systems separated the goals into levels of importance or priority with the more abstract and general goals being worked on first and the more concrete or detailed levels being filled in later. A solution was formed at the most abstract level and the lower, more detailed, levels were then planned using the pre-set skeleton plan formed at the upper levels. No backtracking to the higher levels was possible. Later systems (e.g., NONLIN, Tate 1977) treated the abstraction levels as a guide to a skeleton solution, but were able to re-plan or consider alternatives at any level if a solution could not be found, or if it was indicated that a higher level choice was faulty.

Other hierarchical systems use the abstraction levels as one guide to the ordering of goals, but have other mechanisms that can be considered alongside this. Some systems are able to determine when a particular choice (at whatever level) is sufficiently constrained to be a preferable goal to work on at any time (e.g., MOLGEN, Stefik 1981a,b). A relatively recent approach has included attempting to build models which can plan at different levels of the hierarchy concurrently. A mathematical analysis of some properties which allow systems to assume independence of level effects in some cases has been performed (Yang, 1989).

A second aspect of handling conjunctive goal planning is the treatment of the interactions that arise between the different goals in the planning problem. As an example, consider the situation (from Sacerdoti, 1977) of trying to paint the ceiling and also painting the ladder used for painting the ceiling. If the planner paints the ladder first, the ladder will be wet and the execution agent will be unable to paint the ceiling. Further, getting the paint for the ladder and getting the paint for the ceiling should be combined into a single task. Thus, to get an optimal solution the planner must partially merge the two separate plans (see Figure 4).

Figure 4 goes about here.

Planners can be categorized according to the way in which they manage interactions between goals and the way in which they structure operators in a plan. In terms of goal interactions, there are planners which make the so-called "linearity assumption", that is, that solving one goal (out of a conjunction) and then following this with the solution to the other goals in the conjunction will be successful. This assumption is valid in some domains since the solutions to different goals may often be decoupled. Most current planners do not make the linearity assumption and can consider arbitrary interleavings of all goals and subgoals. These planners are often referred to as "non-linear" planners† as they do not require the linearity assumption to guarantee correct solution†.

Early planning systems would solve the conjunctive goals sequentially and then make a simple check to see if the conjunction of goals still held (e.g., in STRIPS (Fikes & Nilsson, 1971). Where the linearity assumption failed, this could lead to mutual goal or subgoal interference which would often require redundant actions being put into the plan. In fact, in the worst case the planner could get into an endless cycle of re-introducing and trying to satisfy the same goal over and over again.

Several systems followed this which could handle some of the problems introduced when the problems being solved involved interacting goals. These systems either allowed alternate orderings of entire solution paths (HACKER, Sussman 1977), allowing an interfering action (just being introduced to satisfy some goal) to be placed at different (earlier) points in the current plan until a position was found where there is no interaction (WARPLAN, Warren 1974) or regressing the goal (i.e. considering it earlier) when an interaction with some particular solution failed (Waldinger, 1975). (This latter had the advantage that redundant actions were not reintroduced if the

† The term "non-linear" is sometimes used, confusingly, to refer not to this assumption, but rather to the partial ordering of plan steps that some planners use in attempting to handle the problem. We address this issue later in this section.

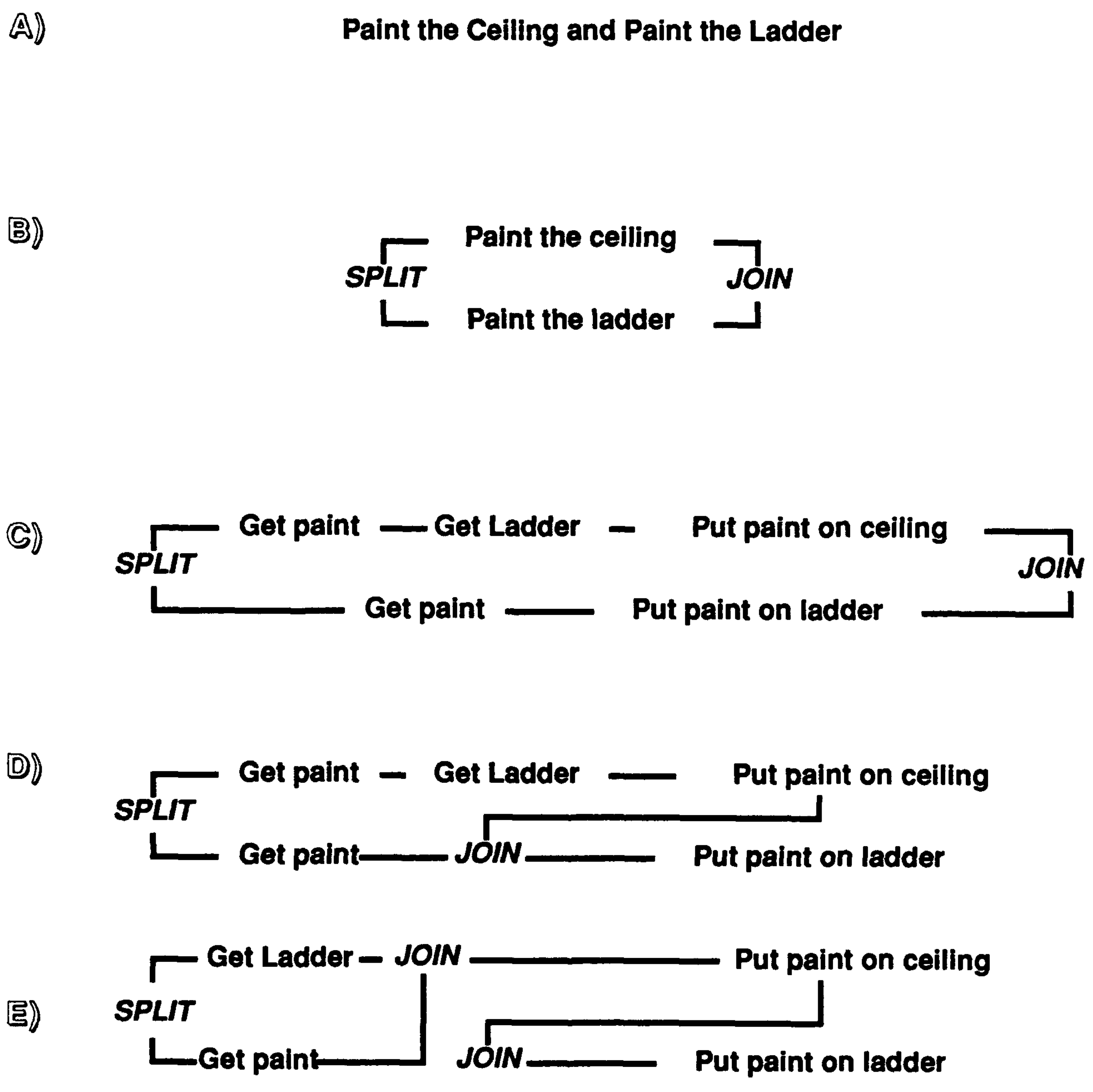


Figure 4: Painting the ceiling (Based on Sacerdoti, 1977).
(A) The conjunctive goal (B) is split into two separate tasks. (C) Plans for each task are generated separately and (D) ordering links are introduced to avoid interactions. Finally, (E) common subgoals are merged.

goal was achievable earlier in the plan).

INTERPLAN (Tate, 1975a) introduced a representation called "Goal Structure" to record the link between an effect of one action that was a precondition (subgoal) of a later one. This representation was additional to the ordering links between the actions themselves (as some actions have effects that are used much later in the plan — Sussman (1973) referred to this as the plan's "teleology.") Interactions were detected as an interference between some new action or goal being introduced and one or more previous goal structures. A minimum set of goalreorderings could be suggested that corrected for the interactions found. This approach was found to be more efficient as it considered fewer re-orderings of the given goals and subgoals than the earlier methods.

NOAH (Sacerdoti, 1975), introduced code called "critics" which were used to search for interactions between parts of the plan, which was represented as a *partial ordering*. This was an important innovation as it allowed the planner to use a "least commitment" strategy — separate operators would be considered unordered with respect to each other unless a critic introduced an ordering relation to avoid an interaction (as was done in Figure 4d). NOAH used a data structure, the table of Multiple Effects (TOME), to record the effects of the operators and thus aid in discovering interactions.

NOAH (Sacerdoti, 1977) is often referred to as the first "non-linear" planner —— the term being used in this context to refer to the partially ordered plan structures employed by the system. This is often confused with the fact that NOAH was able to solve problems where the linearity assumption didn't hold. To help avoid this confusion we will not refer to this type of system as non-linear, but rather, since such planners *use* partially ordered plans we will use the term "partial order planners." Thus, NOAH was the first partial-order planner.

NONLIN (Tate, 1975b; 1977) was a partial-order planner modeled on NOAH. As in NOAH, the minimum set of orderings necessary to resolve interactions could be suggested by the introduction of ordering links into a partial order plan when they became essential. The NONLIN system, however, could also consider alternatives if failures on any chosen search branch occurred. NONLIN introduced the notion of a Goal-State Table (GOST) which was a datastructure which could be used to record dependencies between plan steps facilitating the analysis of interactions. Similar analyses of a representation of the effect and condition structure of a plan to detect and correct for interactions have been included in PLANX-10 (Sridharan & Bresina, 1985), SIPE's "Plan Rationale" (Wilkins, 1983) and Dean's (1985) work.

A further refinement of the analysis of interactions was added to the DEVISER system (Vere, 1983). When individual goals or actions in a plan had time constraints on when they could be satisfied or performed, the detection and correction of interactions were sensitive to the temporal displacement introduced . This helped limit the number of legal solutions that could be proposed. DEVISER allowed for planning in the presence of deadlines and external events. In addition, the use of objects being manipulated as scarce resources on which usage conflicts occur and need to be avoided was incorporated in the MOLGEN (Stefik, 1981a) and SIPE (Wilkins, 1983) planners. Viewing time as such a resource is an active topic being considered in current research (cf. Dean & Boddy, 1987).

5. Operator Representation

The very first systems to handle planning problems simply selected appropriate operators to apply to a problem by considering the "differences" between the goal state and the initial state and looking for operators which were known to remove such differences. GPS (Newell & Simon, 1963), for example, directly associated the operators for the problem with the "differences" they could reduce. Thus, an operator such as "PAINT(Robot,x)" would be associated with achieving the fact "PAINTED(x)."

STRIPS (Fikes and Nilsson, 1971) used this notion of differences, as well as ideas from the situation calculus (McCarthy & Hayes 1969) and Green's (1969) QA3 program, to make the assumption that the initial world model would only be changed by a set of additions to and deletions from the statements modelling the world state — everything else remaining unchanged. (This is sometimes called the "STRIPS assumption.") STRIPS then defined an operator as having an add-list, a delete-list and a preconditions-formula (to define the applicability or sub-goaling conditions). Operators were selected on the basis of the recognition of goals which matched statements on the add-lists (i.e. those statements the operator could add to the current state).

For non-primitive actions, Sacerdoti's (1975) NOAH system used procedurally specified "SOUP" (Semantics Of User's Problem) code to introduce appropriate methods of achieving goals or orderings to correct for interactions into the network of actions. Later planners introduced declarative representations to handle this, with operators which were extensions to the STRIPS operator type of formalism (e.g., NONLIN'S Task Formalism (Tate, 1977) and SIPE Notation (Wilkins, 1983)). As well as add and delete lists and precondition formulae, an "expansion" of the operator to a lower level of detail could be specified as a partial order on suitable sub-actions and sub-goals.

More recent systems have continued to add more information to the operators. One, of particular importance, has been adding information about resource usage to the operators, so that planners can reason about limited resources. Time constraints have also been encoded on operators. DEVISER (Vere, 1983), for example, provided a method for specifying a time window on goals and activities, for external events and their time of occurrence, and for delayed events caused some time after a planned action. NONLIN+ (Tate & Whiter, 1984) added the capability of representing multiple limited resources and making selection from appropriate activities on the basis of reducing some overall computed "preference" between them. The definition of shared objects as resources and the declaration of the use of such resources in operators was also provided in SIPE (Wilkins, 1983).

Airplan (Masui et al, 1983) maintained information on the operators so as to be able to reason about time intervals and about how concurrent actions in these intervals could interact. O-Plan (Bell & Tate, 1985) uniformly represents time constraints (and resource usage) by a numeric *(min, max)* pair which bound the actual values for activity duration, activity start and finish times, and delays between activities. The actual values may be uncertain for various reasons, such as the plan being at a high abstraction level, not having chosen values for objects referred to in the plan or uncertainty in modelling the domain. Constraints can be stated on the time and resource values which may lead to the planner finding that some plans in its search space are invalid. EXCALIBUR (Drabble, 1988) allowed for planning in the face of external continuous processes which were modelled qualitatively.

As the domains being considered by planning systems have become more dynamic (see next section) information has also been placed on operators to allow for execution time monitoring (Firby, 1989) , for bounded computation or real-time scheduling needs (Kaelbling, 1987; Hendler, 1989), to allow for different add- and delete-lists depending on execution time success or failure, and to predict the probability of success of an operator (Miller et.al., 1985) (See Figure 5).

Figure 5 Goes around here.

```
PICKUP-NEAREST(LOC,X)

FILTER CONDITION:
        For-all (X,Y) DIST(LOC,X) < DIST(LOC,Y)
            & OBTAINABLE(X)
PRECONDITION:  ONTABLE(X) ^
                HANDEMPTY ^
                CLEAR(X)
MONITOR: (NEW-OBJ-MON
    Queue-length 1
      Run-time 75
    Run-every 500
    Reports:
    if DIST(LOC,newobj) < DIST(LOC,X)
        then UPDATE(X = NewObj)
        (PRESERVE-OBJ-MON
    Queue-length 1
    Run-time 25
    Run-every 250
    Reports:
    if NOT(PRESERVE(x)) then FAILURE)
STEPS: OPERATOR-MOVE-TO(X)
  LIFT(X)
FAILURE-DEL: (KNOWN (LOC X))
FAILURE-ADD: (Achieve
(PICKUP-NEAREST(CurLoc,X)))
SUCCESS-ADD: AT (X)
          HOLDING(X)
SUCCESS-DEL: OBTAINABLE(X)
          ONTABLE(X)
          HANDEMPTY
          CLEAR(X)
PROBABILITY: .72
RESOURCES: TIME(DIST(LOC,X) AVG-VEL)
   Consumes(Arm)
```

Figure 5: An operator for a (hypothetical) modern planning system.
The operator is annotated to include preconditions, filter conditions, execution time monitoring (with real-time scheduling constraints), steps to be accomplished, add- and delete-lists for success and failure, resource usage information, and a success probability.

6. Planning and Execution

Most of the systems described so far have assumed that the planner is possessed of complete knowledge of the current state of the world and the cause–and–effect relationships that govern changes in that world. Clearly this approach is in need of revision in situations where separate execution cannot be guaranteed to succeed. This can occur when agents outside of the control of the planner may cause changes in the environment or where the planner may be uncertain of information that can only be ascertained while the plan is being run.

For example, the STRIPS system (Fikes et. al., 1972a) was used to plan the motion of a robot called *Shakey* and to control it as it pushed a set of boxes through a number of interconnecting rooms. A well known SRI film showed *Shakey* following a STRIPS generated plan using an execution monitor called PLANEX. Charley Rosen, the SRI AI Lab founder, dressed in a sinister cloak, appeared and disrupted the position of the boxes during execution. PLANEX was able to make use of information maintained by STRIPS to recover.

This approach to dealing with failure when it arises has come to be known as "replanning," and is typically assumed to occur when a planner recognizes a mismatch between the expected and actual state of the world. Hayes (1975) proposed that the subgoal trees and decision graphs used in the formation of the plan could be used to guide replanning. Daniel (1983) explored the use of a similar mechanism in a partial order planner. The PRIAR system (Kambhampati 1989; Kambhampati & Hendler, 1989) uses a well–defined subset of this information, known as the validation structure, to provide such guidance. (A formal treatment of replanning can be found in Morgenstern 1987).

An alternative to replanning is to actually expect potential failures and to plan for them. This can involve planning for expected possibilities, such as waiting for a light to turn green before crossing a street (Drummond, 1986), or it can involve scheduling monitoring tasks — tests to be run at execution time, associated with fixes to be used in the case of failed tests (cf. Doyle, et. al, 1986). An example of this (Kaelbling, 1987) would be having a robot check to see if two walls were equidistant at some point during the traversal of a hallway. If not, the robot could achieve this equidistance, and then continue. Schoppers (1987) has proposed taking disjunctive planning to an extreme by generating "universal" plans — plans with conditional tests to deal with all possible execution–time contigencies.

Another approach to handling change in the environment is to provide a different sort of integration between generation and execution of plans. McDermott's (1978) NASL system, for example, interleaved plan generation and execution by choosing one step at a time and executing it. This made the planner more susceptible to errors caused by interactions, but made it less susceptible to errors caused by change in the environment. This works well in environments where a small amount of change may occur, but is insufficient in domains in which rapid reaction is needed (for example the Traffic World (Hendler & Sanborn, 1987) shown in Figure 6). This approach has been extended by Drummond (1989), where *Situated Control Rules* are used to inform an independently competent execution system — the execution system can, if necessary, act without a plan; the plan simply serves to increase the system's goal–achieving abilities.

Figure 6 goes about here.

Much recent work has dealt with designing mechanisms which can handle rapidly changing environments. Most of this work achieves responsiveness by giving up complex planning for shallow planning or by planning using tightly coupled sensing and action (Rosenschein, 1982; Chapman & Agre, 1987; Sanborn & Hendler, 1988). Some of this work, however, has dealt directly with the issues of how to map from planning to reaction. Firby (1989) proposes "reactive action packages" which essentially replace the operators in the system with procedures that include a

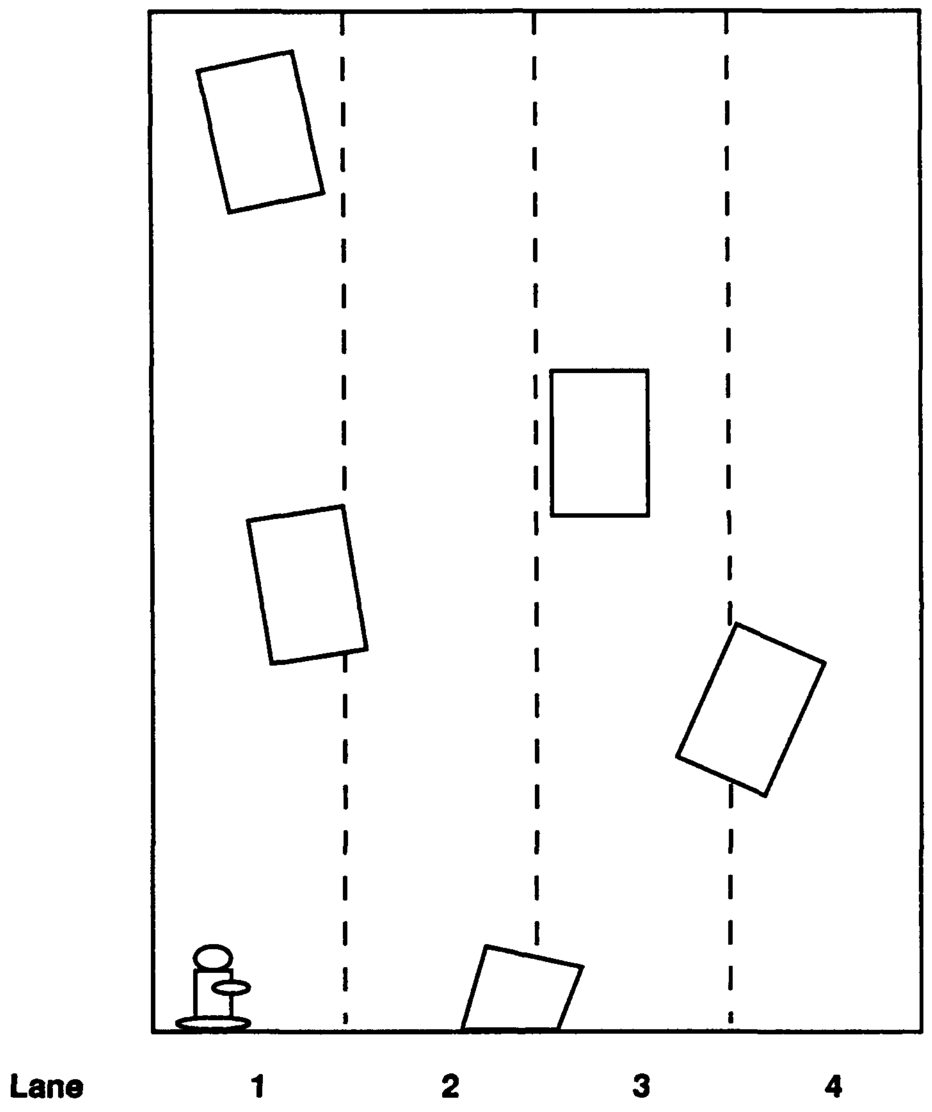


Figure 6: The Traffic World (Hendler & Sanborn, 1987).
The Traffic World consists of a straight stretch of "4–lane highway" along which "cars" travel at various speeds. There is a much slower moving "robot" at one end of the highway, who is trying to get across to the opposite side. Cars changes lanes and speeds as they proceed.

reactive component. Kaelbling's and Rosenschein's (1988) GAPPS system is a compiler that translates constraint expressions into directly executable circuits for use in robotic control systems. Georgeff and Lansky (1987) describe the use of a meta-reasoning system which can choose from a variety of execution time options based on the goals being pursued by the system. Ambros-Ingerson and Steel (1987) propose an approach to integrating planning and execution for changing domains using an agenda-driven control structure in which actions which are serially initiated can run concurrently, with information acquiring actions (for monitoring the environment) included. In addition, an effort is being made to extend temporal representations to handle simultaneously occuring events and event interactions. The use of these extended representations for planning is discussed by Pednault (1987).

In addition, it now appears that an important part of planning in dynamic domains involves making tradeoffs — specifically trading precision in decision making for time in responding to events. In the last few years, a number of researchers have attempted to improve the responsiveness of planning systems operating in dynamic domains by directly reasoning about the utility of planning (Dean, 1987; Horvitz, 1988; Russell & Wefald, 1989). This work has involved an examination both of reasoning about these trade-offs during plan generation (Kanazawa and Dean, 1989; Heckerman et al, 1989) or during execution (Boddy and Dean, 1989; Horvitz et al, 1989).

7. Learning and Memory

The concentration in most of the work on planning has been on generating plans from scratch, not learning from experience. Thus, much of the classical work in planning has been "ahistoric" — that is, asked to solve the same problem again the planner performs no better than it did the first time. Recently, due both to the gains being made in machine learning and the new work on case-based reasoning, designing planning systems which learn from experience has been an option.

The earliest approach to learning in plans was the MACROPs (for macro-operators) work of (Fikes et al 1982b) which extended STRIPS to do some limited learning from its failures. When a portion of a plan was found to have succeeded, the entire set of operators could be turned into a single operator whose preconditions and effects were identical with the preconditions and effects of the operator sequence. The operators were generalized by using variables to replace the constants found in the specific solution from which the new operator had been derived. Minton's (1985) Prodigy/EBL system used a similar approach, but applied an explanation-based learning algorithm to provide more accurate generalizations.

Case-based reasoning approaches to planning have also been attempted. In these systems an old plan is chosen and modified to run in the new situation. Many of these systems have concentrated on guiding the search for the old plan (e.g. JULIA, Kolodner 1987; CHEF, Hammond 1986) and then use a fairly simple mapping to produce the new plan. PLEXUS (Alterman, 1988) used information about the new context to guide the reuse of an existing plan. The PRIAR reuse system (Kambhampati, 1989) applied a case-based approach in the classical planning framework. PRIAR, an extension to NONLIN (Tate, 1977), allowed the planner to annotate plans being created with information about the dependency structure between operators. This information was then used to guide retrieval, reuse, and replanning.

8. Assorted Shorts

As well as the issues discussed above, many other issues arise in the design of planning systems. Some examples include:

Planning with Conditionals and Iterators

Most of the search space control techniques, goal ordering and interaction correction mechanisms developed in AI planners to date have been oriented towards the generation of plans which are fully or partially ordered sequences of primitive actions. However, there has been some effort on the generation of plans which contain conditionals ("if ... then ... else ...") and iterators ("repeat ... until ..."). Conditionals were handled in WARPLAN-C (Warren, 1976) by branching the plan at the conditional and performing separate planning on the two branches using the assumption that the statement in the conditional was true in one branch and false in the other. This led to a case analysis of the separate branches to produce a plan that was tree-structured. Drummond (1985) designed an extension to procedural nets allowing for disjunction and iteration and usable for predicting the behavior of the system where these were present.

Uncertainty in Planning

One source of execution time problems is that of uncertainty which exists during plan generation. If the planner cannot model the real world with complete information, but instead uses some sort of probability-based model, it must deal with low probability events that may occur during execution. (This problem arises in a very significant way for systems which use real sensors to perceive the world). Work using probability estimates during plan generation includes Dean et. al (1989) and Dean & Kanazawa (1989). Segre (1988) examines the issue of execution time failures of plans based on uncertain information.

Distributed Planning

Some systems have been exploring distributing sources of problem solving expertise or knowledge during planning. They allow fully distributed planning with the sub-problems being passed between specialized planning experts. The use of the experts, or the behaviors they may perform, are controlled through a centralized blackboard and executive (with a system rather like priority scheduling of parallel processes) or may be controlled in a more distributed fashion via pairwise negotiation. Examples of relevant work include Smith's (1977) Contract Net, Corkill (1979), Kornfeld (1979), Konolige and Nilsson (1980) Georgeff (1982), and Corkill and Lesser (1983).

9. Recommended Reading

A good and reasonably up-to-date account of AI planning techniques and systems is given in Charniak and McDermott's (1985) *Introduction to Artificial Intelligence* textbook. In particular, chapter 9 and sections of chapters 5 and 7 are relevant. Somewhat earlier material is provided in Elaine Rich's (1983) *Artificial Intelligence* textbook. Nilsson's (1980) book on the *Principles of Artificial Intelligence* provides a uniform treatment of planning techniques available up to the time it was published. There are several useful summaries of early AI planning work in the *Handbook of Artificial Intelligence* (Barr & Feigenbuam, 1981) volume I section II.D and volume III sections XI.B, XI.C and XV. A collection of recent papers concerning planning can be found in Georgeff and Lansky's (1987) *Reasoning about Actions and Plans.* A collection of previously published papers on planning and reasoning about actions will soon be available as *Readings in Planning* (Allen et. al.) , expected to be published in the Spring of 1990.

Allen, J., Hendler, J. and Tate, A. (eds.) (1990) "Readings in Planning," Los Altos, Ca: Morgan-Kaufmann. Barr, A. and Feigenbaum, E.A. (1981) "The Handbook of Artificial Intelligence" Los Angeles, Ca: William Kaufmann.

Charniak. E. and McDermott, D.V. (1985) "Introduction to Artificial Intelligence," Reading, Ma: Addison-Wesley.

Georgeff, M. and Lansky, A. (eds.) (1987) "Reasoning about actions and plans," Los Altos, Ca: Morgan-Kaufmann

Nilsson, N.J. (1980) "Principles of Artificial Intelligence", Palo Alto, Ca: Tioga Press.

Rich, E. (1983) "Artificial Intelligence", McGraw-Hill, New York.

10. Summary

Planning systems have been an active research topic within Artificial Intelligence for nearly thirty years. There have been a number of techniques developed during that period which still form an essential part of many of today's AI planning systems. In this paper we have tried to provide a broad coverage of the major ideas in the field of AI planning, and have attempted to show the direction in which current research is going. Such a task is never-ending, and any finite document must per force be incomplete. We have provided references to connect each idea to the appropriate literature and to allow readers immediate access to the papers most relevant to their research or applications work.

Acknowledgements

We are grateful for input from research colleagues active in AI planning who have helped us structure this review. We hope that we have not misrepresented anyone in showing how the research themes, techniques and planning systems relate to one another. We also apologize to colleagues for simplifications made in our descriptions of past planning systems. Simplification has been a necessary evil for reasons of space; for more complete descriptions the reader should consult the associated references.

The work of Austin Tate and the Planning Group at AIAI is supported by the UK Science and Engineering Research Council (under IED grant 1320, GR/F 36545) and the United States Air Force (under contract EOARD-88-0044). Jim Hendler is also affiliated with the Systems Research Center and the UM Institute for Advanced Computer Studies. His work is funded in part by the US Office of Naval Research grant N00014-88-K-0560 and NSF grant IRI-8907890. A portion of Mark Drummond's work at the NASA Ames Research Center is funded by the Air Force Office of Scientific Research, Artificial Intelligence Research Program.

11. References

Where names of planning systems and other AI software are described in the paper, the appropriate bibliography entries are annotated with the name of the system in square brackets.

Allen, J.F. and Koomen, J.A. (1983) "Planning Using a Temporal World Model", Proceedings of the Eighth International Joint Conference on Artificial Intelligence, 741-747. Karlsruhe, West Germany: International Joint Conferences on Artificial Intelligence. [TIMELOGIC]

Altermann, R. (1988) "Adaptive Planning," Cognitive Science, 12. [PLEXUS]

Ambros-Ingerson, J. and Steel, S., "Integrating planning, execution, and monitoring", In Proceedings of the Eighth National Conference on Artificial Intelligence, 83-88. Los Altos, Ca: Morgan-Kaufmann.

Appelt, D.E. (1985) "Planning English Referring Expressions", Artificial Intelligence, 26,1-33. [KAMP]

Bell, C.E. and Tate, A. (1985) "Using Temporal Constraints to Restrict Search in a Planner", Proceedings of the Third Workshop of the Alvey IKBS Programme Planning Special Interest Group, Sunningdale, Oxfordshire, UK, April 1985. Available through the Institute of Electrical Engineers, London, UK [O-PLAN]

Boddy, M. and Dean, T. (1989) "Solving Time-Dependent Planning Problems" Proceedings of the Eleventh International Joint Conference on Artificial Intelligence-89, 979-984. Menlo Park, Ca: Inernational Joint Conferences on Artificial Intelligence

Bresina, J.L. (1981) "An Interactive Planner that Creates a Structured Annotated Trace of its Operation", Rutgers University, Computer Science Research Laboratory, Report CBM-TR-123. [PLANX-10]

Chapman, D. (1985) Nonlinear planning: a rigorous reconstruction. In Proceedings of the Ninth International Joint Conference on Artificial Intelligence, 1022-1024. Menlo Park, Ca: Inernational Joint Conferences on Artificial Intelligence [TWEAK]

Chapman, D. (1987) "Planning for Conjunctive Goals," Artificial Intelligence (32), 1987, 333-377. [TWEAK]

Chapman, D. and Agre, P. (1987) "Abstract Reasoning as Emergent from Concrete Activity" in Georgeff, M. and Lansky, A. (eds.) Reasoning about Actions and Plans, Los Altos, Ca: Morgan-Kaufmann.

Corkill, D.D. (1979) "Hierarchical Planning in a Distributed Environment", Proceedings of the Sixth International Joint Conference on Artificial Intelligence 168–175, Menlo Park, Ca: International Joint Confererces on Artificial Intelligence.

Corkill, D.D. and Lesser, V.R. (1983) "The Use of Meta-level Control for Coordination in a Distributed Problem Solving Network", Proceedings of the Eighth International Joint Conference on Artificial Intelligence, 748–756. Menlo Park, Ca: International Joint Conferences on Artificial Intelligence.

Currie, K. and Tate, A. (1985) "O-Plan - Control in the Open Planner Architecture" BCS Expert Systems Conference. UK: Cambridge University Press. [OPLAN]

Daniel, L. (1983). "Planning and Operations Research", in "Artificial Intelligence: Tools, Techniques and Applications", NY: Harper and Row. New York. [NONLIN]

Davis, R. and Smith, R. (1983) "Negotiation as a Metaphor for Distributed Problem Solving", Artificial Intelligence, 20,63–109.

Davis, P.R. and Chien, R.T. (1977) "Using and Re-using Partial Plans", Proceedings of the Fifth International Joint Conference on Artificial Intelligence. Menlo Park, Ca: International Joint Conferences on Artificial Intelligence.

Dean, T. (1985) "Temporal Reasoning Involving Counterfactuals and Disjunctions", Proceedings of the ninth International Joint Conference on Artificial Intelligence. Menlo Park, Ca: International Joint Conferences on Artificial Intelligence. [TNMS]

Dean, T. "Intractability and time-dependent planning" in Georgeff, M. and Lanksy, A. (eds) Reasoning about Actions and Plans, Los Altos, CA:Morgan-Kaufmann

Dean, T. and Boddy, M. (1987) "Reasoning about Partially Ordered Events," Artificial Intelligence 36,375–399

Dean, T., Firby, J., and Miller, D., (1989) Hierarchical Planning Involving Deadlines, Travel Time and Resources Computational Intelligence (3). [FORBIN]

Dershowitz, N. (1985) "Synthetic programming", Artificial Intelligence, 25, 323–373.

Doran, J.E. and Michie, D. (1966) "Experiments with the Graph Traverser Program" Proceedings of the Royal Society,235–259. [GRAPH TRAVERSER]

Doran, J.E. and Trayner, C. (1985) "Distributed Planning and Execution - Teamwork 1", Computer Science Technical Report, University of Essex, UK [TEAMWORK]

Doyle, J. (1979) "A Truth Maintenance System", Artificial Intelligence, 12, 231–272.

Doyle, R.J., Atkinson, D.J., and Doshi, R.S. (1986) "Generating perception requests and expectations to verify the execution of plans" Proceedings of the Fifth National Conference on Artificial Intelligence, Los Altos, Ca: Morgan-Kaufmann.

Drabble, B. (1988) "Planning and Reasoning with Processes" The Eighth Workshop of the Alvey Planning Special Interest Group, Institute of Electrical Engineers, Nottingham, UK,25–40.

Drummond, M. (1985). Refining and extending the procedural net. Proceedings of Ninth

International Joint Conference on Artificial Intelligence, 1010–1012. Menlo Park, Ca: International Joint Conferences on Artificial Intelligence.

Drummond, M. (1989) Situated Control Rules, Proceedings of Conference on Principles of Knowledge Representation and Reasoning, Toronto, Canada.

Drummond, M., and K.W. Currie (1988) Exploiting Temporal Coherence in Non-linear Plan Construction. Computational Intelligence 4(4).

Drummond, M., and K.W. Currie (1989) Goal Ordering in Partially Ordered Plans. Proceedings of the Eleventh International Joint Conference on Artificial Intelligence, Menlo Park, Ca: International Joint Conferences on Artificial Intelligence..

Duffay, P. and Latombe, J-C (1983) "An Approach to Automatic Robot Programming Based on Inductive Learning", IMAG, Grenoble, France. [TROPIC]

Erman, L.D., Hayes-Roth, F., Lesser, V.R. and Reddy, D.R. (1980) "The HEARSAY-II Speech-understanding System: Integrating Knowledge to Resolve Uncertainty", ACM Computing Surveys, 12, No.2.

Fahlman, S.E. (1974) "A Planning System for Robot Construction Tasks" Artificial Intelligence, 5,1–49.

Faletti, J. (1982) "PANDORA - A Program for Doing Commonsense Reasoning Planning in Complex Situations", Proceedings of the Second National Conference on Artificial Intelligence. Los Altos, Ca: Morgan-Kaufmann. [PANDORA]

Fikes, R.E. (1970) "REF-ARF: A System for Solving Problems stated as Procedures", Artificial Intelligence", 1,27–120.

Fikes, R.E. (1982) "A Commitment-based Framework for Describing Informal Cooperative Work", Cognitive Science, 6,331–347.

Fikes, R.E. and Nilsson, N.J. (1971) "STRIPS: a New Approach to the Application of Theorem Proving to Problem Solving", Artificial Intelligence, 2,189–208. [STRIPS]

Fikes, R.E., Hart, P.E. and Nilsson, N.J. (1972a) "Learning and Executing Generalised Robot Plans", Artificial Intelligence, 3. [STRIPS/PLANEX]

Fikes, R.E., Hart, P.E. and Nilsson, N.J. (1972b) "Some New Directions in Robot Problem Solving", in "Machine Intelligence 7", Meltzer, B. and Michie, D., eds. UK:Edinburgh University Press. [STRIPS]

Firby, J. (1989) "Adaptive Execution in Complex Dynamic Worlds," Doctoral Dissertation, Department of Computer Science, Yale University. [RAPS]

Fox, M.S., Allen, B. and Strohm, G. (1981) "Job Shop Scheduling: an Investigation in Constraint-based Reasoning", Proceedings of the Seventh International Joint Conference on Artificial Intelligence British Columbia. Menlo Park, Ca: International Joint Conferences on Artificial Intelligence. [ISIS-II]

Georgeff, M. (1982) "Communication and Interaction in Multi-agent Planning Systems", Proceedings of the 3rd National Conference on Artificial Intelligence, Los Altos, Ca: Morgan-Kaufmann.

Georgeff, M. and Lansky, A. (1985) "A Procedural Logic", Proceedings of the Ninth International Joint Conference on Artificial Intelligence. Menlo Park, Ca: International Joint Conferences on

Artificial Intelligence.

Georgeff, M, and Lansky, A. (1987) "Reactive Reasoning and Planning" Proceedings of Sixth National Conference on Artificial Intelligence. Los Altos, Ca: Morgan-Kaufmann. [PRS]

Green, C.C. (1969) "Theorem Proving by Resolution as a basis for Question Answering" in Machine Intelligence 4, eds. Meltzer, B. and Michie, D. UK: Edinburgh University Press.

Gupta, N. and Nau, D. (1990) "Optimal Block's World Solutions are NP-Hard," University of Maryland, Computer Science Dept., Technical Report (in press).

Hammond, K. (1986) "Chef: A model of case-based planning" Proceedings of Fifth National Conference on Artificial Intelligence. Los Altos, Ca: Morgan-Kaufmann. [CHEF]

Hart, P., Nilsson, N., and Raphael, B. (1968) "A formal basis for the heuristic determination of minimum cost paths", IEEE Transactions System Science and Cybernetics, SSC-4(2), 100-107. [A*]

Hayes, P.J. (1975) "A Representation for Robot Plans", Advance papers of the 1975 International Joint Conference on Artificial Intelligence, Tbilisi, USSR.

Hayes-Roth, B. and Hayes-Roth, F. (1979) "A Cognitive Model of Planning", Cognitive Science,275-310. [OPM]

Hayes-Roth, B. (1983a) "The Blackboard Architecture: A General Framework for Problem Solving?", Heuristic Programming Project Report No. HPP-83-30. Stanford University.

Hayes-Roth, B. (1983b) "A Blackboard Model of Control", Heuristic Programming Project Report No. HPP-83-38. Stanford University. [OPM]

Heckerman, D., Breese, J. and Horvitz, E. (1989) "The Compilation of Decision Models" 1989 Workshop on Uncertainty in Artificial Intelligence, Windsor, Ontario162-173

Hendler, J.A. (1987) "Integrating Marker-passing and Problem Solving: A spreading activation approach to improved choice in planning" Norwood, NJ: Lawrence Erlbaum Associates. [SCRAPS]

Hendler, J. and Sanborn, J. (1987) "Planning and reaction in dynamic domains" Proceedings DARPA Workshop on Planning, Austin, Tx.

Hendler, J. "Real-time Planning," (1989) AAAI Symposium on Planning and Search, Stanford, Ca.

Hendrix, G. (1973) "Modelling Simultaneous Actions and Continuous Processes", Artificial Intelligence, 4,145-180.

Horvitz, E. (1988) "Reasoning Under Varying and Uncertain Resource Constraints," Proceedings of the Seventh National Conference on Artificial Intelligence, 111-116. Los Altos, Ca: Morgan-Kaufmann.

Horvitz, E. and Cooper, G. and Heckerman, D. (1989) "Reflection and Action Under Scarce Resources: Theoretical Principles and Empirical Study" Proceedings of the Eleventh International Joint Conference on Artificial Intelligence. 1121-1127. Menlo Park, Ca: International Joint Conferences on Artificial Intelligence.

Kaelbling, L. (1987) "An Architecture for Intelligent Reactive Systems" in Georgeff, M. and Lanksy, A. (eds) Reasoning about Actions and Plans, Los Altos, Ca: Morgan-Kaufmann.

Kaelbling, L.P. (1988) Goals as Parallel Program Specifications. Proceedings of the Seventh National Conference on Artificial Intelligence 60–65. Los Altos, Ca: Morgan-Kaufmann. [GAPPS]

Kahn, K. and Gorry, G.A. (1977) "Mechanizing Temporal Knowledge" Artificial Intelligence, 9,87–108.

Kambhampati, S. (1989) Flexible Reuse and Modification in Hierarchical planning: a Validation Structure Based Approach, Doctoral dissertation, Dept. of Computer Science, University of Maryland. [PRIAR]

Kambhampati, S. and Hendler, J. (1989), "Flexible Reuse of Plans via Annotation and Verification", Proceedings of Fifth IEEE Conference on Applications of Artificial Intelligence, Miami, Florida. [PRIAR]

Kanazawa, K. and Dean, T., (1989) "A Model for Projection and Action" Proceedings of the Eleventh International Joint Conference on Artificial Intelligence, 985–99. Menlo Park, Ca: International Joint Conferences on Artificial Intelligence.

Kolodner, J. (1987) "Case–based Problem Solving," 4th International Workshop on Machine Learning, UC Irvine. [JULIA]

Konolige, K. (1983) "A Deductive Model of Belief", Proceedings of the Eighth International Joint Conference on Artificial Intelligence, 377–381. Menlo Park, Ca: International Joint Conferences on Artificial Intelligence.

Konolige, K. and Nilsson, N.J. (1980) "Multi–agent Planning Systems", Proceedings of the First National Conference on Artificial Intelligence, 138–142: Los Altos, Ca: Morgan–Kaufmann.

Korf, R.E., (1987) "Planning as Search: A Quantitative Approach," Artificial Intelligence, (33), 1987, 65–88.

Kornfeld, W.A. (1979) "ETHER: a Parallel Problem Solving System" Proceedings of the Sixth International Joint Conference on Artificial Intelligence. 490–492. Menlo Park, Ca: International Joint Conferences on Artificial Intelligence.

Lansky, A.L. (1987) "A Representation of Parallel Activity Based on Events, Structure, and Causality," in Reasoning About Actions and Plans, Proceedings of the 1986 Workshop at Timberline, Oregon, M. Georgeff and A. Lansky (editors), Los Altos Ca: Morgan Kaufmann Publishers [GEMPLAN]

Lansky, A.L. (1988) "Localized Event–Based Reasoning for Multiagent Domains," Computational Intelligence Journal, Special Issue on Planning, Volume 4, Number 4. [GEMPLAN]

Latombe, J-C. (1976) "Artificial Intelligence in Computer–aided Design – The TROPIC System", Stanford Research Institute AI Center Technical Note 125, Menlo Park, Ca., USA.

Lenat, D.B. (1975) "BEINGS: Knowledge as Interacting Experts", Proceedings of the Fourth International Joint Conference on Artificial Intelligence. 126–133. Menlo Park, Ca: International Joint Conferences on Artificial Intelligence. [PUP]

London, P. (1977) "A Dependency–based Modelling Mechanism for Problem Solving", Dept of Computer Science, University of Maryland, Memo. TR–589.

Luckham, D.C. and Buchanan, J.R. (1974) "Automatic Generation of Programs Containing Conditional Statements", AISB Summer Conference, University of Sussex, UK,102–126.

Masui,S.. McDermott, J. and Sobel, A. (1983) "Decision-Making in Time Critical Situations", Proceedings of the Eighth International Joint Conference on Artificial Intelligence, 233-235. Menlo Park: International Joint Conferences on Artificial Intelligence [AIRPLAN]

McCarthy, J. and Hayes, P.J. (1969) "Some Philosophical Problems from the Standpoint of Artificial Intelligence", in Machine Intelligence 4, Meltzer, B. and Michie, D. eds. Edinburgh: Edinburgh University Press.

McDermott, D.V. (1978) "Planning and Acting" Cognitive Science, 2. [NASL]

McDermott, D.V. (1982) "A Temporal Logic for Reasoning about Processes and Plans", Cognitive Science, 6, 101-155.

McDermott, D.V. and Doyle, J. (1979) "An Introduction to Non-monotonic Logic", Proceedings of the Sixth International Joint Conference on Artificial Intelligence 562-567. Menlo Park: International Joint Conferences on Artificial Intelligence

Mellish, C.S. (1984) "Towards Top-down Generation of Multi-paragraph Text", Proceedings of the Sixth European Conference on Artificial Intelligence, Pisa, Italy. 229.

Miller, D., Firby, J, and Dean, T. (1985) "Deadlines, travel time, and robot problem solving." Proceedings of the Ninth International Joint Conference on Artificial Intelligence. 1052-1054. Menlo Park: International Joint Conferences on Artificial Intelligence [FORBIN]

Minton, S. (1985) "Selectively Generalizing Plans for Problem-Solving" Proceedings of the Ninth International Joint Conference on Artificial Intelligence, 596-599 Menlo Park: International Joint Conferences on Artificial Intelligence [Prodigy/EBL]

Moore, R. (1980) "Reasoning about Knowledge and Action", Technical Report Report No. 191. SRI AI Center.

Morgenstern, L. (1987) "Replanning" Proceedings DARPA Knowledge-Based Planning Workshop, Austin, Tx.

Mostow, D.J. (1983) "A Problem Solver for Making Advice Operational", Proceedings of the Third National Conference on Artificial Intelligence,179-283. Los Altos, Ca: Morgan-Kaufmann.

Nau, D., Yang, Q., and Hendler, J. (1989), "Planning for Multiple Goals with Limited Interactions," Proceedings of Fifth IEEE Conference on Applications of Artificial Intelligence, Miami, Florida.

Newell, A. and Simon, H.A. (1963) "GPS: a Program that Simulates Human Thought", in Feigenbaum, E.A. and Feldman, J. eds. Computers and Thought, New York:McGraw-Hill. [GPS]

Nilsson, N.J. (1971) Problem Solving Methods in Artificial Intelligence New York: McGraw-Hill.

Pednault, E. (1987) Solving Multi-agent dynamic world problems in the classical planning framework, in Georgeff, M. and Lanksy, A. (eds) Reasoning about Actions and Plans. Los Altos, CA: Morgan-Kaufmann

Reiger, C. and London, P. (1977) "Subgoal Protection and Unravelling During Plan Synthesis", Proceedings of the Fifth International Joint Conference on Artificial Intelligence: Menlo Park: International Joint Conferences on Artificial Intelligence

Rich, C. (1981) "A Formal Representation for Plans in the Programmer's Apprentice", Proceedings of the Seventh International Joint Conference on Artificial Intelligence, 1044-1052. Menlo Park: International Joint Conferences on Artificial Intelligence

Rich, C., Shrobe, H.E. and Waters, R.C. (1979) "Overview of the Programmer's Apprentice", Proceedings of the Sixth International Joint Conference on Artificial Intelligence, 827-828. Menlo Park: International Joint Conferences on Artificial Intelligence

Rosenschein, S.J. (1980) "Synchronisation of Multi-agent Plans", Proceedings of the Second National Conference on Artificial Intelligence, Los Altos, Ca: Morgan-Kaufmann..

Rosenschein, S.J. (1981) "Plan Synthesis: A Logical Perspective", Proceedings of the Seventh International Joint Conference on Artificial Intelligence, Menlo Park: International Joint Conferences on Artificial Intelligence.

Rosenschein, S. (1982) "Synchronization of Multi-agent Plans", Proceedings of the Second National Conference on Artificial Intelligence. Los Altos, Ca: Morgan-Kaufmann.

Russell, S.. and Wefald, E. (1989) "Principles of Metareasoning" Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning, Los Altos, Ca: Morgan-Kaufmann

Sacerdoti, E.D. (1973) "Planning in a Hierarchy of Abstraction Spaces", Advance papers of the Third International Joint Conference on Artificial Intelligence. [ABSTRIPS]

Sacerdoti, E.D. (1975) "The Non-linear Nature of Plans", Advance papers of the Fourth International Joint Conference on Artificial Intelligence. [NOAH]

Sacerdoti, E.D. (1977) "A Structure for Plans and Behaviour", Amsterdam: Elsevier-North Holland. [NOAH]

Sacerdoti, E.D. (1979) "Problem Solving Tactics", Proceedings of the Sixth International Joint Conference on Artificial Intelligence. Menlo Park: International Joint Conferences on Artificial Intelligence

Sanborn, J. and Hendler, J. (1988) Monitoring and Reacting: Planning in dynamic domains. International Journal of AI and Engineering, 3(2).

Sathi, A., Fox, M.S. and Greenberg, M. (1985) "Representation of Activity Knowledge for Project Management", IEEE Special Issue of Transactions on Pattern Analysis and Machine Intelligence. [CALLISTO]

Schank, R.C. and Abelson, R.P. (1977) "Scripts, Plans, Goals and Understanding", Hillsdale, NJ: Lawrence Erlbaum Press.

Schoppers, M. (1987) "Universal plans for reactive robots in unpredictable domains," Proceedings of the tenth International Joint Conference on Artificial Intelligence, Menlo Park: International Joint Conferences on Artificial Intelligence

Segre, A. (1988) Machine Learning od Robot Assembly Plans, Norwell, MA:Kluwer Academic Publishers.

Siklossy, L. and Roach, J. (1973) "Proving the Impossible is Impossible is Possible: Disproofs based on Hereditary Partitions", Proceedings of the Third International Joint Conference on Artificial Intelligence. Menlo Park: International Joint Conferences on Artificial Intelligence [DISPROVER/LAWALY]

Siklossy, L. and Dreussi, J. (1975) "An Efficient Robot Planner that Generates its own Procedures", Proceedings of the Third International Joint Conference on Artificial Intelligence. Menlo Park: International Joint Conferences on Artificial Intelligence [LAWALY]

Simmons, R. and Davis, R. (1987) "Generate, Test and Debug: Combining Associational Rules and Causal Models," Proceedings of the Tenth International Joint Conference on Artificial Intelligence, 1071–1078. Menlo Park: International Joint Conferences on Artificial Intelligence [G–T–D]

Smith, R.G. (1977) "The Contract Net: a Formalism for the Control of Distributed Problem Solving", Proceedings of the Ffth International Joint Conference on Artificial Intelligence, 472. Menlo Park: International Joint Conferences on Artificial Intelligence

Smith, R.G. (1979) "A Framework for Distributed Problem Solving", Proceedings of the Sixth International Joint Conference on Artificial Intelligence, Menlo Park: International Joint Conferences on Artificial Intelligence

Sridharan, A. and Bresina, J.L. (1985) "Knowledge structures for planning in realistic domains." *Computers and Mathematics with Applications (Special Issue on Knowledge Representation).* Vol. 11(5), 457–480. [PLANX-10]

Stallman, R.M. and Sussman, G.J. (1977) "Forward Reasoning and Dependency Directed Backtracking", Artificial Intelligence, 9, 135–196.

Steele, G.L. and Sussman, G.J. (1978) "Constraints", MIT AI Lab Memo 502. AI Laboratory, MIT.

Stefik, M.J. (1981a) "Planning with Constraints", Artificial Intelligence, 16,111–140. [MOLGEN]

Stefik, M.J. (1981b) "Planning and Meta-planning", Artificial Intelligence, 16,141–169. [MOLGEN]

Sussman, G.A. (1973) "A Computational Model of Skill Acquisition", M.I.T. AI Lab. Memo no. AI-TR-297, AI Laboratory, MIT [HACKER]

Sussman, G.A. and McDermott, D.V. (1972) "Why Conniving is Better than Planning", MIT AI Lab. Memo 255A. AI Laboratory, MIT [CONNIVER]

Tate, A. (1975a) "Interacting Goals and Their Use", Proceedings of the Fourth International Joint Conference on Artificial Intelligence.215–218 ICJAI* [INTERPLAN]

Tate, A. (1975b) "Project Planning Using a Hierarchical Non-linear Planner", Dept. of Artificial Intelligence Report 25, Edinburgh University. [NONLIN]

Tate, A. (1977) "Generating Project Networks", Proceedings of the Fifth International Joint Conference on Artificial Intelligence, Boston, Ma., USA. [NONLIN]

Tate, A. (1984a) "Planning and Condition Monitoring in a FMS", International Conference on Flexible Automation Systems", Institute of Electrical Engineers, London, UK July 1984. [NONLIN]

Tate, A. (1984b) "Goal Structure: Capturing the Intent of Plans", European Conference on Artificial Intelligence, Pisa, Italy. [NONLIN]

Tate, A. and Whiter, A.M. (1984) "Planning with Multiple Resource Constraints and an Application to a Naval Planning Problem", First Conference on the Applications of Artificial Intelligence, Denver, Colorado, USA. Los Altos, Ca: Morgan-Kaufmann.. [NONLIN+]

Vere, S. (1983) "Planning in Time: Windows and Durations for Activities and Goals", IEEE Trans. on Pattern Analysis and Machine Intelligence, PAMI-5, No. 3, 246–267. [DEVISER]

Vilain, M.B. (1980) "A System for Reasoning about Time", Proceedings of the Second National

Conference on Artificial Intelligence. Los Altos, Ca: Morgan-Kaufmann.

Waldinger, R. (1975) "Achieving Several Goals Simultaneously", SRI AI Center Technical Note 107, SRI, Menlo Park, Ca., USA.

Warren, D.H.D. (1974) "WARPLAN: a System for Generating Plans", Dept. of Computational Logic Memo 76. Artificial Intelligence, Edinburgh University. [WARPLAN]

Warren, D.H.D. (1976) "Generating Conditional Plans and programs", Proceedings of the AISB Summer Conference, University of Edinburgh, UK,344-354 [WARPLAN-C]

Wilensky, R. (1978) "Understanding Goal-based Stories", Dept. of Computer Science, Yale University, Research Report No. 140.

Wilensky, R. (1981a) "Meta-planning: Representing and Using Knowledge about Planning in Problem Solving and Natural Language Understanding", Cognitive Science, 5,197-233.

Wilensky, R. (1981b) "A Model for Planning in Complex Situations", Electronics Research Lab. Memo. No. UCB/ERL M81/49, University of California, Berkeley, Ca., USA.

Wilensky, R. (1983) "Planning and Understanding", Reading, Ma: Addison-Wesley.

Wilkins, D.E. and Robinson, A.E. (1981) "An Interactive Planning System", SRI Technical Note 245. [SIPE]

Wilkins, D.E. (1983) "Representation in a Domain-Independent Planner", Proceedings of the Eighth International Joint Conference on Artificial Intelligence, 733-740 Menlo Park: International Joint Conferences on Artificial Intelligence [SIPE]

Wilkins, D.E. (1988) Practical Planning - Extending the classical AI planning paradigm. Los Altos, Ca: Morgan-Kaufmann.

Yang, Q. (1989) Improving the Efficiency of Planning, Doctoral dissertation, Dept. of Computer Science, University of Maryland.