Extending Conventional Planning Techniques to Handle Actions with Context-Dependent Effects

Edwin P.D. Pednault
Knowledge Systems Research Department
AT&T Bell Laboratories
Holmdel, NJ 07733

ABSTRACT

This paper presents a method of solving planning problems that involve actions whose effects change according to the situations in which they are performed. The approach is an extension of the conventional planning methodology in which plans are constructed iteratively by scanning for goals that are not yet satisfied, inserting actions to achieve them, and introducing additional subgoals to be achieved. The necessary extensions to this methodology to handle context-dependent effects are presented from a general, mathematically rigorous standpoint.

1. INTRODUCTION

Most domain-independent planners synthesize plans via an iterative process of scanning for goals that are not yet satisfied, inserting actions to achieve them, and introducing additional (sub)goals to be achieved. This methodology was originally developed under the assumption that one would be dealing with actions that produce the same effects in every situation [e.g., Fikes and Nilsson 1971; Sussman 1973; Sacerdoti 1973, 1977; Siklóssy and Dreussi 1973; Warren 1974, 1976; Tate 1975, 1977; Vere 1983; Chapman 1987]. However, the methodology has been extended in certain limited ways so as to handle actions with context-dependent effects [e.g., Waldinger 1977; Rosenschein 1981; Kautz 1982; Wilkins 1984, 1987; Schoppers 1987]. Also, a number of domain-specific planners that employ the methodology are able to deal with context-dependent effects within their domains of expertise [e.g., Fahlman 1974; Stefik 1981; Simmons and Davis 1987]. This raises the following question: how can the methodology be fully generalized to treat actions with context-dependent effects? This paper presents a mathematically rigorous analysis of precisely this question.

With context-dependent effects, one must take into account the fact that an action might achieve a goal in certain situations but not in others. To account for this, the analysis introduces the notions of *primary* and *secondary preconditions*. Primary preconditions are simply the usual preconditions for the execution of actions [e.g., McCarthy 1969; Fikes and Nilsson 1971]. Secondary preconditions, on the other hand, define the contexts in which these actions produce particular effects. It is shown that by introducing the appropriate secondary preconditions as subgoals to actions in addition to the primary preconditions, conventional planning techniques can be extended to handle actions with context-dependent effects. It is further shown that these secondary preconditions can be constructed automatically from regression operators.

In addition to introducing secondary preconditions as subgoals, the analysis demonstrates that, to achieve generality, changes must be made in the way plans are modified. To ensure that all solutions to a planning problem can be found, one must *explicitly* consider the possibility of using a single action to achieve several goals simultaneously, as well as the possibility of achieving a goal by preventing it from becoming false. Achieving goals by always introducing new actions is not sufficient.

2. REPRESENTING ACTIONS AND GOALS

For the purposes of the analysis, the standard state-transition model of action will be adopted. In the general form of this model [e.g., Rosenschein 1981; Kautz 1982; Pednault 1987], the world is viewed as being in one of a potentially infinite number of states. The effect of an action is to cause the world to make a transition from one state to another. Each action is therefore modeled by a set of current-state/next-state pairs of the form $\langle s, t\rangle$, where s and t are states, s being the "current-state" and t being the "next-state." This set specifies what the effects of the action would be in each state of the world in which the action can be performed.

For complex domains, the number of states needed to represent a problem may either be infinite or at least so large as to make it impractical to enumerate them all. For such problems, states and actions are usually dealt with indirectly through language.

For this analysis, the only properties that will be required of the language for describing states is that there be no ambiguity as to which states satisfy a given description, that descriptions can be negated, and that they can be conjoined through the use of an 'and' connective. These requirements are consistent with the representations used in most planning systems. They are also consistent with formulas of first-order logic, which will be used in the examples of this paper. For convenience, the terminology of first-order logic will also be adopted; in particular, state descriptions will be called formulas. However, this form of representation is not a requirement for the application of the analysis presented here.

Regression operators [Waldinger 1977; Nilsson 1980; Rosenschein 1981; Kautz 1982] will be used to represent the effects of actions. Although many other formalisms exist, it turns out that the additional subgoals needed to fully extend the conventional planning methodology can be constructed from regression operators. A regression operator is a function that provides descriptions of what has to be true immediately before an action is performed in order for a given condition to be true immediately afterward.

Definition 1. A regression operator a^{-1} for an action a is a function from formulas to formulas with the property that, for every formula φ and every pair of states $\langle s,t\rangle \in a$, if $a^{-1}(\varphi)$ is true in state s, then φ is true in state t.

Using regression operators, we can determine whether a desired condition will be true after executing a sequence of actions. If Γ is a description of what is true prior to executing the sequence of $a_1 \cdots a_n$, and if

$$\Gamma \models a_1^{-1} \circ \cdots \circ a_n^{-1}(\varphi) \tag{1}$$

then φ will be true after execution. In Condition 1, "$\circ$" denotes function composition (i.e., $f \circ g(x) = f(g(x))$) while $\models$ denotes logical implication (i.e., the right-hand side is true in every state in which the left-hand side is true). Obviously, one application of regression operators is in verifying that a plan is correct. The sequence $a_1 \cdots a_n$ is executable and achieves goals G if the following conditions are met:

$$\Gamma \models \pi^{a_1} \tag{2a}$$
$$\Gamma \models a_1^{-1} \circ \cdots \circ a_k^{-1}(\pi^{a_{k+1}}) \quad \text{for all } k,\ 1 \le k < n \tag{2b}$$
$$\Gamma \models a_1^{-1} \circ \cdots \circ a_n^{-1}(G) \tag{2c}$$

where π^{a_i} is a formula describing the preconditions for the execution of action a_i (i.e., π^a is true in state s if and only if $\langle s,t\rangle \in a$ for some state t).

3. THE EXTENDED METHODOLOGY

With the conventional planning methodology, one begins with the empty plan and incrementally modifies it until a complete plan is obtained. At each stage, the partially developed plan is analyzed for goals that are not yet satisfied. The appropriate actions for achieving them are then inserted, producing a new partial plan and initiating a new cycle of analysis and modification. To extend this methodology to handle actions with context-dependent effects, we can make use of the following theorem:

Theorem 1. A condition φ will be true at a point p during the execution of a plan if and only if one of the following holds:

(1) An action is executed prior to point p that causes φ to become true and φ remains true until at least point p.

(2) φ is true in the initial state and remains true until at least point p.

This theorem relies only upon two properties inherent in the state-transition model. It therefore applies to all actions for which this model is appropriate, including those with context-dependent effects. The first property is that the state of the world can change only as the result of an action. Consequently, if a condition φ is true at a point p during the execution of a plan but not at an earlier point, then at some point in between an action must have been performed that made it true. The other property is that plans must be finite. This further implies that φ might become true and then false numerous times, but there must be a last action that finally achieves φ prior to p. This fact is reflected in the first clause of Theorem 1. The second clause reflects the fact that if φ is true at point p and there are no previous points at which it is false, then φ must be true at all points prior to p. The theorem may be stated in terms of regression operators using Condition 1 in the last section as the criterion for determining whether φ is true at a point in a plan. The proof then follows by induction on the number of actions in the plan [Pednault 1986, 1988].

The first clause of Theorem 1 tells us that one way to achieve a goal is for there to be an action in the final plan that causes it to become true. The conventional planning methodology assumes that if a goal has not yet been satisfied, the action that achieves it must be inserted into the plan. However, because plans are constructed incrementally, the action that achieves the goal might already appear in the current partially-constructed plan. Thus, a second way of modifying the plan is to establish the appropriate conditions to enable an existing action to achieve the goal. The second clause of the theorem tells us that a third way of achieving a goal is to prevent it from becoming false if it happens to be true initially.

The following corollary to Theorem 1 provides a formal rationale for the three ways described above of incrementally modifying a plan to achieve a goal. Using this corollary, it can be shown that every solution to a planning problem can be constructed through a combination of inserting new actions to achieve goals, enabling existing actions to achieve goals, and preventing goals from becoming false.

Corollary 1. A condition φ will be true at a point p during the execution of the final plan if and only if one of the following holds:

(1) There exists an action in the final plan prior to point p such that
  (a) The action already appears in the current partial plan.
  (b) The action causes φ to become true in the final plan and φ remains true until at least point p.

(2) There exists an action in the final plan prior to point p such that
  (a) The action does not yet appear in the current plan and must be inserted.
  (b) The action causes φ to become true in the final plan and φ remains true until at least point p.

(3) φ is true in the initial state and remains true until at least point p in the final plan.

Planning can be viewed as a process of asserting which of the clauses of Corollary 1 should hold for each of the goals in one's evolving plan. This entails asserting that certain actions are intended to achieve certain goals while preserving certain others. With actions that produce the same effects in all situations, these assertions amount to verifying that each action achieves and preserves the appropriate goals. Verification is not sufficient, however, when the effects are context dependent, since an action might achieve or preserve a goal in some situations but not in others. It is therefore necessary to assert that the action is carried out in an appropriate context.

In conventional planners, subgoals are used to define the context in which an action is to be performed. Normally, only the preconditions for execution are introduced as subgoals to ensure the executability of an action. To ensure that an action will achieve or preserve an intended goal, a second set of subgoals must be introduced as well. These additional subgoals will be called *secondary preconditions*, the preconditions of execution being the primary ones. Two types of secondary preconditions are needed: *causation preconditions* and *preservation preconditions*. The expression Σ_φ^a will be used to denote the causation precondition defining the context in which performing action a achieves φ, while $\mathbb{P}_\varphi^a$ will denote the preservation precondition defining the context in which performing action a preserves φ.

The use of subgoals to assert the various clauses of Corollary 1 is justified if we can find definitions for causation and preservation preconditions so that the following theorem and corollary hold:

Theorem 2. A condition φ will be true at a point p during the execution of a plan if and only if one of the following holds:

(1) There is an action a prior to point p such that
  (a) Σ_φ^a is true immediately before executing a.
  (b) $\mathbb{P}_\varphi^b$ is true immediately before the execution of each action b between a and point p.

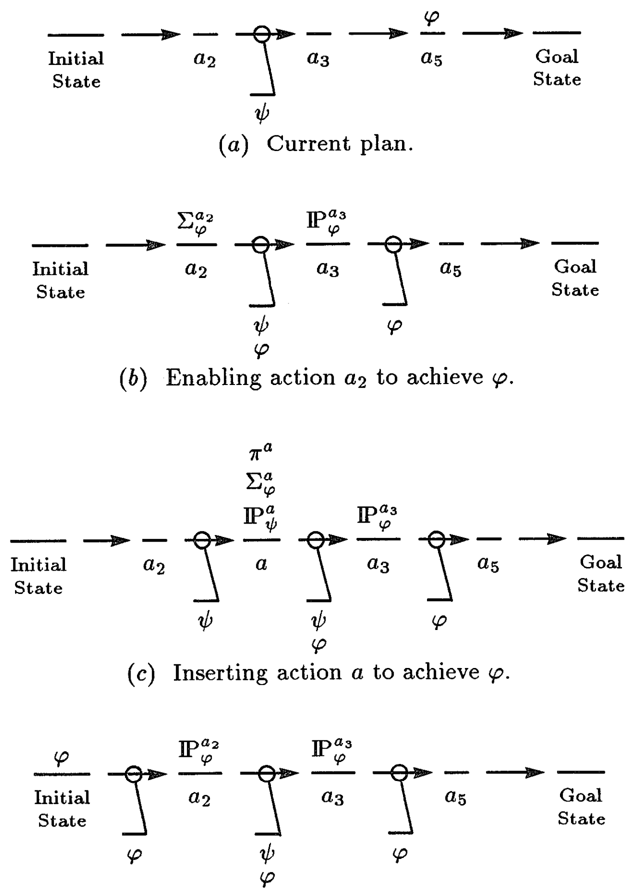
(2) φ is true in the initial state and $\mathbb{P}_\varphi^a$ is true immediately before the execution of each action a prior to point p.

Corollary 2. A condition φ will be true at a point p during the execution of the final plan if and only if one of the following holds:

(1) There exists an action a in the final plan prior to point p such that
  (a) The action already appears in the current plan.
  (b) Σ_φ^a is true immediately before executing a.
  (c) $\mathbb{P}_\varphi^b$ is true immediately before the execution of each action b in the final plan between a and point p.

(2) There exists an action a in the final plan prior to point p such that
  (a) The action does not yet appear in the current plan and must be inserted.
  (b) Σ_φ^a is true immediately before executing a.
  (c) $\mathbb{P}_\varphi^b$ is true immediately before the execution of each action b in the final plan between a and point p.

(3) φ is true in the initial state and $\mathbb{P}_\varphi^a$ is true immediately before the execution of each action a in the final plan prior to point p.



(a) Current plan.
(b) Enabling action a_2 to achieve φ.
(c) Inserting action a to achieve φ.
(d) Protecting φ from the initial state.

Figure 1: Asserting the Clauses of Corollary 2

Corollary 2 provides an explicit mathematical basis for modifying a partially constructed plan to achieve a goal. To illustrate, consider the hypothetical plan shown in Figure 1a. In this plan, φ is a subgoal of action a_5 and ψ is to be protected in the interval between actions a_2 and a_3. Suppose that we wished to use the existing action a_2 to achieve φ. According to the first clause of Corollary 2, this would be accomplished by introducing $\Sigma_\varphi^{a_2}$ as a subgoal to action a_2 to assert that a_2 is to achieve φ, and by introducing $\mathbb{P}_\varphi^{a_i}$ as a subgoal to each action a_i between a_2 and a_5 to assert that these actions are to preserve φ. The resulting plan is shown in Figure 1b.

Suppose instead that we wished to insert a new action a that achieves φ between actions a_2 and a_3. After physically inserting the new action, Σ_φ^a would be introduced as a subgoal to a and $\mathbb{P}_\varphi^{a_i}$ would be introduced as a subgoal to each action a_i between a and a_5, as shown in Figure 1c. However, certain other subgoals must also be introduced. As is usually done, the preconditions for execution π^a must be added to the list of subgoals of a along with the preservation precondition $\mathbb{P}_\psi^a$. The former is needed to ensure that a will be executable in the final plan (see Condition 2 in Section 2). The latter must be introduced as required by Theorem 2, since action a will appear in the final plan between actions a_2 and a_3, and ψ is to be protected in this interval. As this illustrates, it is imperative that a record be kept as to the intervals during which each of the various goals are to be preserved so that the appropriate preservation preconditions can be introduced when actions are inserted into a plan.

Figure 1*d* illustrates the third way of achieving φ by protecting it from the initial state. This requires that the appropriate preservation preconditions be introduced as subgoals to the actions preceding a_5 and that an assertion be made requiring that φ be true in the initial state.

All that remains is to define causation and preservation preconditions in a way that satisfies Theorem 2 and Corollary 2.

4. CAUSATION PRECONDITIONS

To achieve generality, the definition of a causation precondition must take into account that it is not always possible to identify the action in a plan that actually causes a goal to become true. For example, consider the following problem conceived by McDermott and Moore [Moore, personal communication, 1985]. You are placed in a sealed room that contains a bucket of water and two packages, A and B. The packages are identical in every way, except that one contains a time bomb that will explode immediately if opened. The goal is to prevent the bomb from exploding by submerging it in water. The solution, of course, is to place both packages in the bucket. However, it is impossible to tell a priori which action of placing a package in the bucket is actually responsible for immersing the bomb submerged.

To account for the possibility of ambiguity, Σ^a_φ must be a formula that simply defines a context in which φ is guaranteed to be true after performing action a.

> **Definition 2.** A formula Σ^a_φ is a *causation precondition* for action a to achieve φ if and only if for every pair of states $\langle s,t\rangle \in a$, it is the case that if Σ^a_φ is true in state s, then φ is true in state t.

Notice that Σ^a_φ satisfies the definition of a regression operator given in Section 2. To construct causation preconditions, we could therefore let Σ^a_φ be equal to $a^{-1}(\varphi)$:

$$\Sigma^a_\varphi \equiv a^{-1}(\varphi) \ . \tag{3}$$

Although Equation 3 provides a general means of constructing causation preconditions, the formulas produced by this equation can often be strengthened and simplified when it is possible to identify precisely which action actually causes a goal to become true. Σ^a_φ need only satisfy two conditions for Theorem 2 and Corollary 2 to hold in this case: (1) Σ^a_φ must satisfy Definition 2; (2) if φ is currently false and performing a will cause it to become true, then Σ^a_φ must currently be true. These conditions may be written in terms of regression operators as follows:

$$\begin{aligned} \Sigma^a_\varphi &\models a^{-1}(\varphi) \\ \neg\varphi \wedge a^{-1}(\varphi) &\models \Sigma^a_\varphi \ . \end{aligned} \tag{4}$$

Condition 4 can only be used if it is possible to identify precisely which action in a plan actually causes a goal to become true when the plan is executed. This is guaranteed if the truth value of the goal can be ascertained at every point in every executable sequence of actions. The action that causes the goal to become true is then the one for which the goal is false immediately before execution and true immediately after. The requirement that the truth value of the goal be ascertainable may be stated formally in terms of regression operators as follows:

> **Definition 3.** A formula φ is said to be *regressively ascertainable everywhere* with respect to an initial state description Γ and a set of allowable actions $\mathcal{A}$ if and only if the following hold:
>
> (1) $\Gamma \models \varphi$ or $\Gamma \models \neg\varphi$.
>
> (2) For every executable sequence of actions $a_1 \cdots a_n$ drawn from $\mathcal{A}$, where executability is defined by Conditions 2*a* and 2*b* in Section 2, it is the case that $\Gamma \models a_1^{-1} \circ \cdots \circ a_n^{-1}(\varphi)$ or $\Gamma \models a_1^{-1} \circ \cdots \circ a_n^{-1}(\neg\varphi)$.

The term 'regressively' is used to emphasize the fact that the truth value of φ is ascertained by employing regression operators.

5. PRESERVATION PRECONDITIONS

In defining the notion of a preservation precondition for φ, the requirements of Theorem 2 and Corollary 2 permit us to assume that φ has already been made true and we need only establish the appropriate context for it to remain true. $\mathbb{P}^a_\varphi$ may therefore be defined as follows:

> **Definition 4.** A formula $\mathbb{P}^a_\varphi$ is said to be a *preservation precondition* for action a to preserve φ if and only if for every pair of states $\langle s,t\rangle \in a$, if both φ and $\mathbb{P}^a_\varphi$ are true in state s, then φ is true in state t.

While this definition ensures that $\mathbb{P}^a_\varphi$ defines a context in which action a preserves φ, to prove Theorem 2 and Corollary 2 it must also be the case that $\mathbb{P}^a_\varphi$ is true whenever a preserves φ. These two requirements may be characterized in terms of regression operators as follows:

$$\begin{aligned} \mathbb{P}^a_\varphi \wedge \varphi &\models a^{-1}(\varphi) \\ a^{-1}(\varphi) \wedge \varphi &\models \mathbb{P}^a_\varphi \end{aligned} \tag{5}$$

The first part of Condition 5 requires that $\mathbb{P}^a_\varphi$ satisfy Definition 4; the second part ensures that $\mathbb{P}^a_\varphi$ will be true whenever executing a preserves φ.

6. CONCLUSIONS

I have presented a very general analysis of how the conventional approach to synthesizing plans can be extended to handle context-dependent effects. The analysis includes within its scope nondeterministic actions, partial knowledge of the initial state, and arbitrarily complex goals, though these aspects are not fully illustrated in this paper (see [Pednault 1988] for a more thorough discussion). The purpose was to determine how far the conventional planning methodology could be pushed. The class of problems is broad enough to include theorem-proving in first-order logic as a planning problem, where the initial state is a set of axioms, the actions are inference rules, and the goal is the theorem to be proved. This implies that the class as a whole is only partially solvable and, hence, computationally intractable. However, this does not imply that each individual problem or subclass of problems is intractable. The situation is akin to that faced by Waltz in his scene labeling work [Waltz 1975]. The problem of labeling edges in line drawings reduces to graph labeling, which is known to be NP-complete. However, the constraints inherent in real-world scenes often enabled Waltz's program to find a consistent labeling in approximately linear time. Likewise, the constraints inherent in a particular application domain may allow planning problems to be solved in a reasonable amount of time. When applying the results of this paper, the challenge will be to identify the constraints that will lead to efficient planning. Several of the issues that must be addressed to achieve efficiency are discussed in a forthcoming paper [Pednault, 1988].

REFERENCES

[Chapman, 1987] D. Chapman. Planning for conjunctive goals. *Artificial Intelligence*, Vol. 32, No. 3, pp 333–377 (July 1987).

[Fahlman, 1974] S.E. Fahlman. A planning system for robot construction tasks. *Artificial Intelligence*, Vol. 5, pp 1–49 (1974).

[Fikes and Nilsson, 1971] R.E. Fikes and N.J. Nilsson. STRIPS: a new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, Vol 2, pp 189–208 (1971).

[Kautz 1982] H.A. Kautz. *A First-Order Dynamic Logic for Planning.* Masters Thesis, Department of Computer Science, University of Toronto, Toronto, Canada (May 1982).

[McCarthy and Hayes, 1969] J. McCarthy and P. Hayes. Some philosophical problems from the standpoint of artificial intelligence. In *Machine Intelligence 4*, B. Meltzer and D. Michie (eds.), pp 463–502 (Edinburgh University Press, Edinburgh, Scotland, 1969).

[Nilsson, 1980] N.J. Nilsson. *Principles of Artificial Intelligence* (Tioga Publishing Company, Palo Alto, California, 1980).

[Pednault, 1986] E.P.D. Pednault. *Toward a Mathematical Theory of Plan Synthesis*, Ph.D. thesis, Department of Electrical Engineering, Stanford University, Stanford, California (December 1986).

[Pednault, 1987] E.P.D. Pednault. Formulating multiagent, dynamic-world problems in the classical planning framework. In *Reasoning about Actions and Plans: Proceedings of the 1986 Workshop*, M.P. Georgeff and A.L. Lansky (eds.), (Morgan Kaufmann, Los Altos, California, 1987).

[Pednault, 1988] E.P.D. Pednault. Synthesizing plans that contain actions with context-dependent effects. *Computational Intelligence* (to appear, 1988).

[Rosenschein, 1981] S.J. Rosenschein. Plan synthesis: a logical perspective. *Proc. IJCAI 7*, University of British Columbia, Vancouver, Canada, pp 331–337 (August 1981).

[Sacerdoti, 1973] E.D. Sacerdoti. Planning in a hierarchy of abstraction spaces. *Proc. IJCAI-73*, Stanford University, Stanford, California, pp 412–422 (August 1973). Expanded version, *Artificial Intelligence*, Vol. 5, No. 2, pp 115–135 (Summer 1974).

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

[Schoppers, 1987] M.J. Schoppers. Universal plans for reactive robots in unpredictable environments. *Proc. IJCAI-87*, Milan, Italy, pp 1039–1046 (August 1987).

[Siklóssy and Dreussi. 1973] L. Siklóssy and J. Dreussi. An efficient robot planner which generates its own procedures. *Proc. IJCAI-73*, Stanford University, Stanford, California, pp 423–430 (August 1973).

[Simmons and Davis, 1987] R. Simmons and R. Davis. Generate, test and debug: combining associational rules and causal models. *Proc. IJCAI-87*, Milan, Italy, pp 1071–1078 (August 1987).

[Stefik, 1981] M. Stefik. Planning with constraints (MOLGEN: Part 1). *Artificial Intelligence*, Vol. 16, No. 2, pp 111–140 (May 1981).

[Sussman, 1973] G.J. Sussman. A computational model of skill acquisition. Technical Report AI TR-197, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, Massachusetts (August 1973).

[Tate, 1975] A. Tate. Interacting goals and their use. *Proc. IJCAI-75*, Tbilis, Georgia, USSR, pp 215–218 (September 1975).

[Tate, 1977] A. Tate. Generating project networks. *Proc. IJCAI-77*, Massachusetts Institute of Technology, Cambridge, Massachusetts, pp 888–893 (August 1977).

[Vere, 1983] S. Vere. Planning in time: windows and durations for activities and goals. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol 5., No. 3, pp 246–267 (May 1983).

[Waldinger, 1977] R. Waldinger. Achieving several goals simultaneously. In *Machine Intelligence 8*, E. Elcock and D. Michie (eds.), pp 94–136 (Ellis Horwood, Edinburgh, Scotland, 1977).

[Waltz, 1975] D. Waltz. Understanding line drawings of scenes with shadows. In *The Psychology of Computer Vision*, P.H. Winston (ed.), pp 19–91 (McGraw Hill, New York, New-York, 1975).

[Warren, 1974] D.H.D. Warren. WARPLAN: a system for generating plans. Memo No. 76, Department of Artificial Intelligence, University of Edinburgh, Edinburgh, Scotland (June 1974).

[Warren, 1976] D.H.D. Warren. Generating conditional plans and programs. *Proc. AISB Summer Conference*, University of Edinburgh, Edinburgh, Scotland, pp 344–354 (July 1976).

[Wilkins, 1984] D.E. Wilkins. Domain-independent planning: representation and plan generation. *Artificial Intelligence*, Vol. 22, No. 3, pp 269–301 (1984).

[Wilkins, 1987] D.E. Wilkins. Using causal rules in planning. Technical Note 410, Artificial Intelligence Center, SRI International, Menlo Park, California (1987).