# An Average Case Analysis of Planning

**Tom Bylander**
Laboratory for Artificial Intelligence Research
Department of Computer and Information Science
The Ohio State University
Columbus, Ohio 43210
byland@cis.ohio-state.edu

## Abstract

I present an average case analysis of propositional STRIPS planning. The analysis assumes that each possible precondition (likewise postcondition) is equally likely to appear within an operator. Under this assumption, I derive bounds for when it is highly likely that a planning instance can be efficiently solved, either by finding a plan or proving that no plan exists. Roughly, if planning instances have $n$ conditions (ground atoms), $g$ goals, and $O(n\sqrt[g]{\delta})$ operators, then a simple, efficient algorithm can prove that no plan exists for at least $1-\delta$ of the instances. If instances have $\Omega(n(\ln g)(\ln g/\delta))$ operators, then a simple, efficient algorithm can find a plan for at least $1-\delta$ of the instances. A similar result holds for plan modification, i.e., solving a planning instance that is close to another planning instance with a known plan. Thus it would appear that propositional STRIPS planning, a PSPACE-complete problem, is hard only for narrow parameter ranges, which complements previous average-case analyses for NP-complete problems. Future work is needed to narrow the gap between the bounds and to consider more realistic distributional assumptions and more sophisticated algorithms.

## Introduction

Lately, there has been a series of worst-case complexity results for planning, showing that the general problem is hard and that several restrictions are needed to guarantee polynomial time (Bäckström and Klein, 1991; Bylander, 1991; Bylander, 1993; Chapman, 1987; Erol *et al.*, 1992a; Erol *et al.*, 1992b). A criticism of such worst-case analyses is that they do not apply to the average case (Cohen, 1991; Minsky, 1991).

Recent work in AI has shown that this criticism has some merit. Several experimental results have shown that specific NP-complete problems are hard only for narrow ranges (Cheeseman *et al.*, 1991; Minton *et al.*, 1992; Mitchell *et al.*, 1992) and even the problems within these ranges can be efficiently solved (Selman *et al.*, 1992). Theoretical results also support this conclusion (Minton *et al.*, 1992; Williams and Hogg, 1992). However, it must be noted that all this work makes a strong assumption about the distribution of instances, namely that the probability that a given constraint appears in a problem instance is independent of what other constraints appear in the instance.

This paper presents an average-case analysis of propositional STRIPS planning, a PSPACE-complete problem (Bylander, 1991). Like the work on NP-complete problems, I make a strong distributional assumption, namely that each possible precondition (likewise postcondition) is equally likely to appear within an operator, and that the probability of a given operator is independent of other operators. Under this assumption, I derive bounds for when it is highly likely that a planning instance can be efficiently solved, either by finding a plan or proving that no plan exists.

Given that planning instances have $n$ conditions (ground atoms) and $g$ goals, and that operators have $r$ preconditions on average and $s$ postconditions on average, I derive the following results. If the number of operators is at most $((2n-s)/s)\sqrt[g]{\delta}$, then a simple, efficient algorithm can prove that no plan exists for at least $1-\delta$ of the instances. If the number of operators is at least $e^r e^{sg/n}(2n/s)(2+\ln g)(\ln g/\delta)$, then a simple, efficient algorithm can find a plan for at least $1-\delta$ of the instances. If $r$ and $s$ are small, e.g., the number of pre- and postconditions remains fixed as $n$ increases, then these bounds are roughly $O(n\sqrt[g]{\delta})$ and $\Omega(n(\ln g)(\ln g/\delta))$, respectively.

A similar result holds for plan modification. If the initial state or goals are different by one condition from that of another planning instance with a known plan, and if there are at least $e^{r+s}(2n/s)(\ln 1/\delta)$ operators, then it is likely $(1-\delta)$ that adding a single operator converts the old plan into a solution for the new instance.

Thus it would appear that propositional STRIPS planning is hard only for narrow parameter ranges, which complements previous average-case analyses for NP-complete problems. Future work is needed to narrow the gap between the bounds and to consider more

realistic distributional assumptions and more sophisticated algorithms.

The rest of the paper is organized as follows. First, definitions and key inequalities are presented. Then, the average-case results are derived.

## Preliminaries

This section defines propositional STRIPS planning, describes the distribution of instances to be analyzed, and presents key inequalities.

### Propositional STRIPS Planning

An instance of *propositional STRIPS planning* is specified by a tuple $\langle \mathcal{P}, \mathcal{O}, \mathcal{I}, \mathcal{G} \rangle$, where:

$\mathcal{P}$ is a finite set of ground atomic formula, called the *conditions*;

$\mathcal{O}$ is a finite set of *operators*; the preconditions and postconditions of each operator are satisfiable sets of positive and negative conditions;

$\mathcal{I} \subseteq \mathcal{P}$ is the *initial state*; and

$\mathcal{G}$, the *goals*, is a satisfiable set of positive and negative conditions.

Each subset $S \subseteq \mathcal{P}$ is a *state*; $p \in \mathcal{P}$ is true in state $S$ if $p \in S$, otherwise $p$ is false in state $S$. If the preconditions of an operator are satisfied by state $S$, then the operator can be applied, and the resulting state is determined by deleting the negative postconditions from $S$ and adding the positive postconditions (cf. (Fikes and Nilsson, 1971)). A solution plan is a sequence of operators that transforms the initial state into a goal state, i.e., a state that satisfies the goals.

### Distributional Assumptions

Let $n$ be the number of conditions. Let $o$ be the number of operators. Let $r$ and $s$ respectively be the expected number of pre- and postconditions within an operator. Let $g$ be the number of goals.

For given $n$, $o$, $r$, $s$, and $g$, I assume that *random planning instances* are generated as follows:

For each condition $p \in \mathcal{P}$, $p$ is a precondition of an operator with probability $r/n$. If $p$ is a precondition, it is equally likely to be positive or negative. For postconditions, $s/n$ is the relevant probability.

For each condition $p \in \mathcal{P}$, $p \in \mathcal{I}$ (the initial state) with probability .5.

For the goals, $g$ conditions are selected at random and are set so that no goal is satisfied in the initial state. This latter restriction is made for ease of exposition.

It must be admitted that these assumptions do not approximate planning domains very well. For example, there are only $b$ *clear* conditions for a blocks-world instance of $b$ blocks compared to $O(b^2)$ *on* conditions. However, every blocks-world operator refers to one or more *clear* conditions, i.e., a given *clear* condition appears more often within the set of ground operators than a given *on* condition. Also, there are correlations between the conditions, e.g., $clear(A)$ is more likely to appear with $on(A, B)$ than with $on(C, D)$. Similar violations can be found for any of the standard toy domains.

Ultimately, the usefulness of these assumptions will depend on how well the threshold bounds of the analysis classify easiness and hardness of real planning domains. Until then, I shall note that the assumptions are essentially similar to previous work on NP-complete problems as cited in the introduction, but for a different task (planning) in a harder complexity class (PSPACE-complete). Also, the assumptions permit a clean derivation of interesting bounds, which suggest that hard planning instances are localized to a narrow range of the number of operators (the $o$ parameter). Finally, the gap between the assumptions and reality will hopefully spur further work to close the gap.

### Algorithm Characteristics

Each algorithm in this paper is incomplete but sound, i.e., each algorithm returns correct answers when it returns yes or no, but might answer "don't know." Specifically, "success" is returned if the algorithm finds a solution plan, "failure" is returned if the algorithm determines that no plan exists, and "don't know" is returned otherwise.

The performance of a given algorithm is characterized by an accuracy parameter $\delta$, $0 < \delta < 1$. Each result below shows that if the number of operators $o$ is greater than (or less than) a formula on $n$, $r$, $s$, $g$, and $\delta$, then the accuracy of the algorithm on the corresponding distribution (see Distributional Assumptions section) will be at least $1 - \delta$.

### Inequalities

I freely use the following inequalities. For nonnegative $x$ and $y$:

$$
\begin{aligned}
& e^{-x/(1-x)} \leq 1 - x \quad \text{for } 0 \leq x < 1 \\
& 1 - x \leq e^{-x} \\
& x/(1+x) \leq 1 - e^{-x} \\
& 1 - e^{-x} \leq x \\
& xy/(1+xy) \leq 1 - (1-x)^y \quad \text{for } 0 \leq x < 1 \\
& 1 - (1-x)^y \leq xy/(1-x) \quad \text{for } 0 \leq x < 1
\end{aligned}
$$

The first two inequalities are easily derivable from (Cormen *et al.*, 1990). The last four inequalities are derivable from the first two.

## When Plan Nonexistence is Efficient

If there are few operators, it becomes unlikely that the postconditions of the operators cover all the goals, i.e., that some goal is not a postcondition of any operator. This leads to the following simple algorithm:

POSTS-COVER-GOALS
  **for** each goal
    **if** the goal is not a postcondition of any operator
      **then return** failure
  **return** don't know

The following theorem characterizes when POSTS-COVER-GOALS works.

**Theorem 1** *For random planning instances, if* $o \leq ((2n-s)/s)\sqrt[g]{\delta}$, *then* POSTS-COVER-GOALS *will determine that no plan exists for at least* $1-\delta$ *of the instances.*

**Proof:** The probability that there exists a goal that is not a postcondition of any operator can be developed as follows. Consider a particular goal to be achieved:

| | |
|---|---|
| $s/2n$ | probability that an operator achieves the goal[1] |
| $1-s/2n$ | probability that an operator doesn't achieve the goal |
| $(1-s/2n)^o$ | probability that no operator achieves the goal |
| $1-(1-s/2n)^o$ | probability that some operator achieves the goal |
| $(1-(1-s/2n)^o)^g$ | probability that *every* goal is achieved by some operator |

It can be shown that:

$$(1-(1-s/2n)^o)^g \leq (so/(2n-s))^g$$

which is less than $\delta$ if:

$$o \leq ((2n-s)/s)\sqrt[g]{\delta}$$

Thus, if the above inequality is satisfied, then the probability that some goal is not a postcondition of any operator is at least $1-\delta$. □

For fixed $\delta$ and increasing $n$ and $g$, the above bound approaches $(2n-s)/s$. If $s$ is also fixed, the bound is $O(n)$.

Naturally, more complex properties that are efficient to evaluate and imply plan non-existence could be used, e.g., the above algorithm does not look at preconditions. Any algorithm that also tests whether there are postconditions that cover the goals will have performance as good and possibly better than POSTS-COVER-GOALS.

## When Finding Plans is Efficient

With a sufficient number of operators, then it becomes likely that some operator will make progress towards the goal. In this section, I consider three algorithms. One is a simple forward search from the initial state to a goal state, at each state searching for an operator that decreases the number of goals to be achieved. The second is a backward search from the goals to a smaller set of goals to the initial state. The third is a very simple algorithm for when the initial state and goals differ by just one condition.

### Forward Search

Consider the following algorithm:

FORWARD-SEARCH($S$,$O$)
  **if** $\mathcal{G}$ is satisfied by $S$, **then return** success
  **repeat**
    **if** $O$ is empty **then return** don't know
    randomly remove an operator from $O$
  **until** applying an operator satisfies more goals
  let $S'$ be the result of applying the operator to $S$
  **return** FORWARD-SEARCH($S'$,$O$)

If FORWARD-SEARCH($\mathcal{I}, \mathcal{O}$) is called, then each operator in $\mathcal{O}$ is considered one at a time. If applying an operator increases the number of satisfied goals, the current state $S$ is updated. FORWARD-SEARCH succeeds if it reaches a goal state and is noncommittal if it runs out of operators.

FORWARD-SEARCH only considers each operator at most once. I do not propose that this "feature" should be incorporated into practical planning algorithms, but it does simplify the analysis. Specifically, there is no need to consider the probability that an operator has some property given that it is known that the operator has some other property. Despite this handicap, FORWARD-SEARCH is surprisingly robust under certain conditions. First, I demonstrate a lemma for the number of operators that need to be considered to increase the number of satisfied goals.

**Lemma 2** *Consider random planning instances except that* $d$ *of the* $g$ *goals are not satisfied. If at least*

$$e^r e^{s(g-d)/n}(1+2n/sd)(\ln 1/\delta)$$

*operators are considered, then, for at least* $1-\delta$ *of the instances, one of those operators will increase the number of satisfied goals.*

**Proof:** The expression for the probability can be developed as follows:

| | |
|---|---|
| $(1-r/2n)^n$ | probability that a state satisfies the preconditions of an operator, i.e., each of $n$ conditions is not a precondition with probability $1-r/n$; alternatively, a condition is a matching precondition with probability $r/2n$ |
| $(1-s/2n)^{g-d}$ | probability that the postconditions of an operator are consistent with the $g-d$ goals already achieved |

[1] For arithmetic expressions within this paper, multiplication has highest precedence, followed by division, logarithm, subtraction, and addition. E.g., $1-s/2n$ is equivalent to $1-(s/(2n))$.

$(1 - s/2n)^d$ probability that the postconditions do not achieve any of the $d$ remaining goals, i.e., for each goal, it is not a postcondition with probability $1 - s/n$; alternatively, it is a precondition of the wrong type with probability $s/2n$

$(1 - (1 - s/2n)^d)$ probability that the postconditions achieve at least one of the $d$ remaining goals

Thus, the probability $p$ that a particular operator can be applied, will not clobber any satisfied goals, and will achieve at least one more goal is:

$$p = (1 - r/2n)^n (1 - s/2n)^{g-d} (1 - (1 - s/2n)^d)$$

$1 - p$ is the probability that the operator is unsatisfactory, and $(1 - p)^o$ is the probability that $o$ operators are unsatisfactory.

If $(1 - p)^o \leq \delta$, then there will be some satisfactory operator with probability at least $1 - \delta$. This inequality is satisfied if $o \geq (1/p)(\ln 1/\delta)$ because in such a case:

$$(1 - p)^o \leq e^{-po} \leq e^{-\ln 1/\delta} = \delta$$

All that remains then is to determine an upper bound on $1/p$, i.e., a lower bound on $p$. For each term of $p$:

$$(1 - r/2n)^n \geq e^{-rn/(2n-r)} \geq e^{-r}$$
$$(1 - s/2n)^{g-d} \geq e^{-s(g-d)/(2n-s)} \geq e^{-s(g-d)/n}$$
$$(1 - (1 - s/2n)^d) \geq sd/(2n + sd)$$

Inverting these terms leads to the bound of the lemma. □

To describe FORWARD-SEARCH, the expression in the lemma must be summed for each $d$ from 1 to $g$, which leads to the following theorem:

**Theorem 3** *For random planning instances, if*

$$o \geq e^r e^{sg/n} (2n/s)(3/2 + \ln g)(\ln g/\delta)$$

*then* FORWARD-SEARCH *will find a plan for at least* $1 - \delta$ *of the instances after considering the above number of operators.*

**Proof:** For $g$ goals, the number of satisfied goals will be increased at most $g$ times. If each increase occurs with probability at least $1 - \delta/g$, then $g$ increases (the most possible) will occur with probability at least $1 - \delta$.

Thus, Lemma 2 can be applied using $\delta/g$ instead of $\delta$. Summing over the $g$ goals leads to:

$$\sum_{d=1}^{g} e^r e^{s(g-d)/n} (1 + 2n/sd)(\ln g/\delta)$$
$$= e^r e^{sg/n} (\ln g/\delta) \left( \left( \sum_{d=1}^{g} e^{-sd/n} \right) + \left( \sum_{d=1}^{g} e^{-sd/n} (2n/sd) \right) \right)$$
$$\leq e^r e^{sg/n} (\ln g/\delta) \left( \left( \sum_{d=1}^{g} e^{-sd/n} \right) + \left( (2n/s) \sum_{d=1}^{g} 1/d \right) \right)$$

For the two sums:

$$\sum_{d=1}^{g} e^{-sd/n} \leq \int_0^g e^{-sx/n} dx \leq n/s$$
$$\sum_{d=1}^{g} 1/d \leq 1 + \int_1^g (1/x) dx = 1 + \ln g$$

which leads to:

$$\left( \sum_{d=1}^{g} e^{-sd/n} \right) + \left( (2n/s) \sum_{d=1}^{g} 1/d \right)$$
$$\leq n/s + (2n/s)(1 + \ln g)$$
$$= (2n/s)(3/2 + \ln g)$$

Combining all terms results in the bound of the theorem. □

The bound is exponential in the expected numbers of pre- and postconditions. Naturally, as operators have more preconditions, it becomes exponentially less likely that they can be applied. Similarly, as operators have more postconditions, it becomes exponentially less likely that the postconditions are consistent with the goals already achieved. Note though that if $g \leq n/s$, then $e^{sg/n} \leq e$, so the expected number of postconditions $s$ is not as important a factor if the number of goals are small.

## Backward Search

Consider the following algorithm for searching backward from a set of a goals:

```
BACKWARD-SEARCH(G,O)
  if G = ∅ then return success
  while O ≠ ∅
    randomly remove an operator from O
    let R and S be its pre- and postconditions
    if G is consistent with S, and
        |(G − S) + R| < |G|
      then return
          BACKWARD-SEARCH((G − S) + R, O)
  return don't know
```

Like FORWARD-SEARCH, BACKWARD-SEARCH makes a single pass through the operators, but in this case, BACKWARD-SEARCH starts with the goals and looks for an operator whose preconditions results in a smaller number of goals. In fact, if BACKWARD-SEARCH succeeds, then it will have discovered a sequence of operators that achieves a goal state from any initial state, although note that the first operator in this sequence (the last operator selected by BACKWARD-SEARCH) must not have any preconditions; otherwise $|(G - S) + R|$ would be non-zero. Having such an operator is probably unrealistic; nevertheless, the results below suggest that reducing a set of goals into a much smaller set of goals is often possible, which, of course, can then be followed by forward search.

I first introduce a lemma for the number of operators needed to find one operator that reduces the number of goals. Space limitations prevent displaying the complete proof.

**Lemma 4** *For random planning instances with $r \leq n/2$ and $s \leq n/2$, if at least the following number of operators are considered:*

$$e^{2r}e^{sg/n}((3n+sg)/sg)(\ln 1/\delta)$$

*then, for $1-\delta$ of the instances, some operator will reduce the number of goals.*

**Proof Sketch:** The following expression gives the probability $p$ that, for a random operator, the preconditions are a subset of the goals, the postconditions are consistent with the goals, and there is one goal equal to a postcondition, but not in the preconditions.

$$p = (1-r/n)^{n-g}((1-r/2n)^g(1-s/2n)^g - (1-r/2n-s/n+(3rs/4n^2))^g)$$

Bounding this expression leads to the bound of the lemma. □

Similar to Theorem 3, this expression needs to be summed for $g$ goals down to 1 goal. This is done to prove the next theorem (proof omitted).

**Theorem 5** *For random planning instances with $r \leq n/2$ and $s \leq n/2$, if*

$$o \geq e^{2r}(3n/s)(4e^{sg/n}+3(\ln g)+3)(\ln g/\delta)$$

*then* BACKWARD-SEARCH *will find a plan for at least $1-\delta$ of the instances after considering the above number of operators.*

Comparing the two bounds for FORWARD-SEARCH and BACKWARD-SEARCH, the bound for BACKWARD-SEARCH is worse in that it has a larger constant and has a $e^{2r}$ term as opposed to a $e^r$ term for the FORWARD-SEARCH bound. Because BACKWARD-SEARCH does not use the initial state, some increase would be expected. However, the BACKWARD-SEARCH bound is better in that one component is additive, i.e., $O(e^{sg/n}+\ln g)$; whereas the corresponding subexpression for the FORWARD-SEARCH bound is $O(e^{sg/n}\ln g)$. The reason is that $e^{sg/n}$ (see Lemma 4) is maximum when $g$ is at its maximum, while the maximum value for $(3n+sg)/sg$ is when $g$ is at its minimum.

Of course, it should be mentioned that rather crude inequalities are used in both cases to derive simplified expressions. A careful comparison of the probabilities derived within the Lemmas would perhaps be a more direct route for comparing the algorithms, but I have not done this yet.

## Plan Modification

So far I have considered the problem of generating a plan from scratch. In many cases, however, the current planning instance is close to a previously solved instance, e.g., (Hammond, 1990; Kambhampati and Hendler, 1992).

Consider a simplified version of plan modification, specifically, when the initial state or set of goals of the current planning instance differs by one condition from a previously solved instance. In this case, the new instance can be solved by showing how the new initial state can reach the old initial state, or how the old goal state can reach a new goal state. Within the framework of random planning instances then, I shall analyze the problem of reaching one state from another when the two states differ by one condition, i.e., there are $n$ goals, and all but one goal is true of the initial state.

The worst-case complexity of this problem, like the problem of planning from scratch, is PSPACE-complete (Nebel and Koehler, 1993). However, the following theorem shows that it is usually easy to solve this problem if there are sufficient operators.

**Theorem 6** *For random planning instances in which there are $n$ goals, where $n-1$ goals are true of the initial state, if:*

$$o \geq e^r e^s(2n/s)(\ln 1/\delta)$$

*then, for at least $1-\delta$ of the instances, some operator solves the instance in one step.*

**Proof:** First, I develop the probability $p$ that a random operator solves a random instance. The probability that the preconditions are consistent with the initial state is $(1-\frac{r}{2n})^n$. The probability that the postconditions are consistent with the $n-1$ goals already achieved is $(1-s/(2n))^{n-1}$. In addition, the probability that the goal to be achieved is a postcondition is $s/(2n)$.[2] Thus:

$$p = (1-r/2n)^n(1-s/2n)^{n-1}s/2n$$

Lower bounds for $p$ are:

$$p \geq e^{-rn/(2n-r)}e^{-sn/(2n-s)}s/2n \geq e^{-r}e^{-s}s/2n$$

The probability that none of $o$ operators solves the instance is $(1-p)^o$. If $o$ satisfies the inequality stated in the theorem, then:

$$(1-p)^o \leq e^{-po} \leq e^{-\ln 1/\delta} = \delta$$

which proves the theorem. □

Thus, for fixed $r$, $s$, and $\delta$, a linear number of operators suffice to solve planning instances that differ by one condition from previously solved instances. So, for at least the distribution of planning instances considered here, plan modification is easier than planning from scratch by roughly $O(\ln^2 g)$.

[2] This does not scale up to the case of attaining $g$ goals by a single operator. The probability that the postconditions of a random operator contain the $g$ goals is $(s/2n)^g$, i.e., exponentially small in the number of goals.

## Remarks

I have shown that determining plan existence for propositional STRIPS planning is usually easy if the number of operators satisfy certain bounds, and if each possible precondition and postcondition is equally likely to appear within an operator, independently of other pre- and postconditions and other operators. Assuming that the expected numbers of pre- and postconditions are fixed, then it is usually easy to show that instances with $n$ conditions and $O(n)$ operators are unsolvable, and it is usually easy to find plans for instances with $n$ conditions, $g$ goals, and $\Omega(n \ln^2 g)$ operators. In addition, plan modification instances are usually easy to solve if there are $\Omega(n)$ operators. The constants for the latter two results are exponential in the expected numbers of pre- and postconditions.

This work complements and extends previous average-case analyses for NP-complete problems. It complements previous work because it suggests that random planning instances are hard only for a narrow range of a particular parameter, in this case, the number of operators. It extends previous work because the worst-case complexity of propositional STRIPS planning is PSPACE-complete, thus, suggesting that PSPACE-complete problems exhibit similar threshold phenomena.

This work also provides theoretical support for reactive behavior. A main tenet of reactive behavior is that sound and complete planning, besides being too inefficient, is often unnecessary, i.e., states can be mapped to appropriate operators without much lookahead. The analysis of the FORWARD-SEARCH algorithm, which only does a limited one-step lookahead, shows that this tenet is true for a large subset of the planning problem.

Further work is needed to narrow the gap between the bounds derived by this paper and to analyze more realistic distributions. In particular, the assumption that pre- and postconditions are independently selected is clearly wrong. Nevertheless, it would be interesting to empirically test how well the bounds of this paper classify the hardness of planning problems.

## References


Allen, J.; Hendler, J.; and Tate, A., editors 1990. *Readings in Planning.* Morgan Kaufmann, San Mateo, California.

Bäckström, C. and Klein, I. 1991. Parallel non-binary planning in polynomial time. In *Proc. Twelfth Int. Joint Conf. on Artificial Intelligence*, Sydney. 268–273.

Bylander, T. 1991. Complexity results for planning. In *Proc. Twelfth Int. Joint Conf. on Artificial Intelligence*, Sydney. 274–279.

Bylander, T. 1993. The computational complexity of propositional STRIPS planning. *Artificial Intelligence.* To appear.

Chapman, D. 1987. Planning for conjunctive goals. *Artificial Intelligence* 32(3):333–377. Also appears in (Allen *et al.*, 1990).

Cheeseman, P.; Kanefsky, B.; and Taylor, W. M. 1991. Where the really hard problems are. In *Proc. Twelfth Int. Joint Conf. on Artificial Intelligence*, Sydney. 331–337.

Cohen, P. R. 1991. A survey of the eighth national conference on artificial intelligence: Pulling together or pulling apart? *AI Magazine* 12(1):17–41.

Cormen, T. H.; Leiserson, C. E.; and Rivest, R. L. 1990. *Introduction to Algorithms.* MIT Press, Cambridge, Massachusetts.

Erol, K.; Nau, D. S.; and Subrahmanian, V. S. 1992a. On the complexity of domain-independent planning. In *Proc. Tenth National Conf. on Artificial Intelligence*, San Jose, California. 381–386.

Erol, K.; Nau, D. S.; and Subrahmanian, V. S. 1992b. When is planning decidable? In *Proc. First Int. Conf. on AI Planning Systems*, College Park, Maryland. 222–227.

Fikes, R. E. and Nilsson, N. J. 1971. STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence* 2(3/4):189–208. Also appears in (Allen *et al.*, 1990).

Hammond, K. J. 1990. Explaining and repairing plans that fail. *Artificial Intelligence* 45(1-2):173–228.

Kambhampati, S. and Hendler, J. A. 1992. A validation-structure-based theory of plan modification and reuse. *Artificial Intelligence* 55(2-3):193–258.

Minsky, M. 1991. Logical versus analogical or symbolic versus connectionist or neat versus scruffy. *AI Magazine* 12(2):34–51.

Minton, S.; Johnston, M. D.; Philips, A. B.; and Laird, P. 1992. Minimizing conflicts: A heuristic repair method for constraint satisfaction and scheduling problems. *Artificial Intelligence* 58:161–205.

Mitchell, D.; Selman, B.; and Levesque, H. 1992. Hard and easy distributions of sat problems. In *Proc. Tenth National Conf. on Artificial Intelligence*, San Jose, California. 459–465.

Nebel, B. and Koehler, J. 1993. Plan modification versus plan generation: A complexity-theoretic perspective. In *Proc. Thirteenth Joint Int. Conf. on Artificial Intelligence*, Chambery, France. To appear.

Selman, B.; Levesque, H.; and Mitchell, D. 1992. A new method for solving hard satisfiability problems. In *Proc. Tenth National Conf. on Artificial Intelligence*, San Jose, California. 440–446.

Williams, C. P. and Hogg, T. 1992. Using deep structure to locate hard problems. In *Proc. Tenth National Conf. on Artificial Intelligence*, San Jose, California. 472–477.