Some Results on the Complexity of Planning with Incomplete Information

Patrik Haslum and Peter Jonsson

Department of Computer and Information Science
Linköping University, S-581 83 Linköping, Sweden
{pahas,petej}@ida.liu.se

Abstract Planning with incomplete information may mean a number of different things; that certain facts of the initial state are not known, that operators can have random or nondeterministic effects, or that the plans created contain sensing operations and are branching. Study of the complexity of incomplete information planning has so far been concentrated on probabilistic domains, where a number of results have been found. We examine the complexity of planning in nondeterministic propositional domains. This differs from domains involving randomness, which has been well studied, in that for a nondeterministic choice, not even a probability distribution over the possible outcomes is known. The main result of this paper is that the non-branching plan existence problem in unobservable domains with an expressive operator formalism is EXPSPACE-complete. We also discuss several restrictions, which bring the complexity of the problem down to PSPACE-complete, and extensions to the fully and partially observable cases.

1 Introduction

Though the planning problem, including the computational complexity of planning, has been long studied in AI, the problem of planning with incomplete information has received relatively little attention, and the computational complexity of this problem even less. Most work has been in the area of Markov decision processes, which model any uncertainty as a probability distribution over the set of alternatives. In contrast, we consider the problem of planning with *nondeterministic* operators; we assume no information about the relative probabilities of different operator outcomes.

1.1 Approaches to Planning with Incomplete Information

There are basically two ways in which incomplete information has been dealt with in planning research.

A *conformant* planner [15] tries to construct plans that will work in every foreseeable case, by choosing operators such that the goal is achieved no matter what the value of an initially unknown state component or the outcome of a nondeterministic operator. When the domain contains sensing operators that can provide information on the state of the world during plan execution, *contingent* planners [13, 12, 3, 14] can create branching plans, plans that execute differently depending on the information sensed.

From the theory of Markov decision processes (MDPs), we borrow the terminology on observability and call a domain *fully observable* if there are sensing operators without preconditions for every proposition in the domain, *unobservable* if there are no sensing operators and *partially observable* when none of the other terms apply. Obviously, in an unobservable domain there is no point in creating a branching plan as there is no information to branch on, and thus the contingent plan existence problem coincides with the conformant version of the problem.

In *probabilistic* planning [9], all uncertainties are assigned a probability distribution. The planner seeks to create a plan that will maximize the probability of achieving the goal. Probabilistic planners may also take advantage of sensing operators to create branching plans [4].

1.2 Nondeterministic Operators in Planning

The problem we consider in this paper is *conformant plan existence* in unobservable domains where we lack even a probability distribution over the possible outcomes of a nondeterministic operator.

Such incompleteness naturally arises in domains where the exact process involved in an action is too complex to model, and where statistical data to create a probabilistic model is lacking. An example of this kind is domains involving agents that act independently of the planning agent, such as a game situation. Another example is synthesis of "fault-tolerant" plans.

1.3 Previous Results on the Complexity of Planning

There is a growing body of work on the complexity of the planning problem. We summarize the results relevant for comparison with the main result of this paper.

Deciding plan existence in a propositional domain with deterministic operators and complete information about the initial state was shown to be PSPACE-complete by Bylander [2]. Erol *et al.* [5] showed that in a general first order domain the same problem is undecidable. Restricted to a first order domain with a finite set of ground terms, the problem is EXPSPACE-complete. This representation has essentially the same expressive power as the propositional, but allows for exponentially more compact encoding. Since they consider the complete information case, all these results naturally concern only non-branching plans.

For planning with operators having context dependent and probabilistic effects, Littman [10] showed that deciding the existence of a branching plan with a probability of success greater or equal to some threshold ϵ in a fully observable propositional domain is EXPTIME-complete.

Many other results on the complexity of probabilistic planning come from the theory of Markov decision processes. The problem of finding an optimal history dependent m-horizon policy for a partially observable, succinctly represented MDP corresponds to the problem of finding an optimal (*i.e.* maximizing the probability of success) branching plan of depth at most m in a partially observable propositional domain with context dependent operators and probabilistic effects. This problem is known to be EXPSPACE-complete [6]. Interestingly, both the unobservable and the fully observable domain versions of the problem are easier, being EXPTIME- and NEXPTIME-complete, respectively.

1.4 Contributions

In this paper, we consider the complexity of the problem of deciding the existence of non-branching plans in an unobservable propositional domain, and show this problem to be EXPSPACE-complete. We also discuss a number of restrictions on problem instances or solutions that bring the complexity down to PSPACE. Finally, we show briefly how a limited form of branching in plans can be encoded into a non-branching plan, taking advantage of the expressive operator formalism.

2 Preliminaries

The following basic definitions are used throughout the paper.

Definition 1. *By a* planning scenario *we mean a tuple*

$$\mathcal{P} = (\mathcal{D}, \mathcal{O}, I, G)$$

where the domain $\mathcal{D}$ *is a set of propositional symbols,* $\mathcal{O}$ *is a set of operators and* I *and* G *are the initial state and goal state descriptions, respectively. The definition of operators varies, depending on the case under consideration.*

A state set description, *or* state formula, *is any boolean combination of literals over* $\mathcal{D}$*. A state is represented by the subset of* $\mathcal{D}$ *consisting of all propositions true in the state. A state formula* σ *denotes a set of possible states, in the obvious way;*

$$mod(\sigma) = \{\xi \subseteq \mathcal{D} | \xi \models \sigma\}$$

where $\xi \models \sigma$ *means that* σ *holds in state* ξ*, according to the standard semantics of propositional logic.*

For a set of states Ξ *and an operator* O *we define operator semantics in terms of the* successor state set function, $S(O, \Xi)$*, denoting* *the set of states that may result from applying O in any state $\xi \in \Xi$. A sequence of operators $O_1; \ldots ; O_n$ constitutes a* plan *for scenario $\mathcal{P}$ iff $S(O_n, S(O_{n-1}, \ldots S(O_1, mod(I)) \ldots)) \models G$.*

The definition of the successor state set function depends on what operator formalism we consider. In this paper, we introduce a kind of operators, called *actions*, that may have nondeterministic and context dependent effects.

Definition 2. *A* reassignment *is an expression $x{:=}v$, where $x \in \mathcal{D}$ and $v \in \{\mathcal{T}, \mathcal{F}\}$. A* condition *is a boolean combination of literals over $\mathcal{D}$. An* action *is*

(i) A reassignment, $x{:=}v$.
(ii) A sequential composition of actions $\alpha_1; \alpha_2$, where α_1 and α_2 are actions.
(iii) A conditional composition of actions $\bigwedge \gamma_i \to \alpha_i$, where each γ_i is a condition and each α_i an action, and the conditions are mutually exclusive.
(iv) A nondeterministic composition of actions $\alpha_1 \vee \alpha_2$, where α_1 and α_2 are actions.

Given a set of possible states Ξ and an action A, the successor state function is defined inductively as

$$\begin{aligned}
S(x{:=}\mathcal{T}, \Xi) &= \{\xi' | \xi \in \Xi \wedge \xi' = \xi \cup \{x\}\} \\
S(x{:=}\mathcal{F}, \Xi) &= \{\xi' | \xi \in \Xi \wedge \xi' = \xi - \{x\}\} \\
S(\alpha_1; \alpha_2, \Xi) &= S(\alpha_2, S(\alpha_1, \Xi)) \\
S(\textstyle\bigwedge \gamma_i \to \alpha_i, \Xi) &= \textstyle\bigcup S(\alpha_i, \{\xi \in \Xi | \xi \models \gamma_i\}) \cup \{\xi \in \Xi | \xi \not\models \bigvee \gamma_i\} \\
S(\alpha_1 \vee \alpha_2, \Xi) &= S(\alpha_1, \Xi) \cup S(\alpha_2, \Xi)
\end{aligned}$$

We use sequential composition in place of conjunction to avoid the ambiguity that may otherwise arise when two subactions affect the same propositions. We may encode the unordered conjunction of two actions α_1 and α_2 as $(\alpha_1; \alpha_2) \vee (\alpha_2; \alpha_1)$[1], or we may include unordered conjunction into the definition of actions; it does not affect the results.

[1] Note, though, that this expression grows exponentially with the number of actions composed.

3 Nondeterminism, Context Dependency and Unobservable Domain

In this section, we present the main result of the paper, that the non-branching plan existence problem for a planning scenario $\mathcal{P} = (\mathcal{D}, \mathcal{O}, I, G)$ with unobservable propositional domain and containing actions is EXPSPACE-complete. In the process, we also show a bound on the length of plans (Lemma 3).

We show hardness through a reduction from the EXPSPACE--complete problem of deciding universality for regular expressions with exponentiation. To clarify the reduction, we first convert the regular expression into a nondeterministic finite automaton augmented with counters of bounded capacity. For definitions and results on regular expressions and finite automata not presented here, see for instance [7].

Definition 3. *A* regular expression with exponentiation *is formed as a normal regular expression from symbols of the alphabet, but using in addition to the operations of concatenation, union and closure also exponentiation, which is written r^n, where r is a regular expression with exponentiation and n a natural number written in binary (the symbols* 1 *and* 0 *are assumed not to belong to the alphabet of the expression). The expression r^n denotes all strings consisting of n consecutive substrings denoted by r, i.e. $L(r^n) = \{w_1 w_2 \dots w_n | w_i \in L(r)\}$.*

For example, the expression $(a^{10} + b^{10})^{10}$ over the alphabet $\{a, b\}$ denotes the set $\{aaaa, aabb, bbaa, bbbb\}$.

When counting the number of operators in a regular expression with exponentiation, r, we include alphabet symbols but count each exponentiaton as one operator, regardless of the number it is raised to. By the length of r, $|r|$, we mean the number of operators in r plus the number of digits in the exponents in r [2]. For example, the expression above consists of 6 operators and its length is 12.

Definition 4. *A* nondeterministic finite automaton with counters*, or NFAC, is a nondeterministic finite automaton augmented with*

[2] This does not exactly correspond the number of symbols in r because concatenation and exponentiation are written without actual symbols for the operators.

a set of counters C. *Each counter* c *is associated with a set of states* $states_c$, *a natural number* $limit_c$, *and the two states* $loop_c$ *and* $continue_c$. *Initially, all counters are set to* 0. *When the automaton enters any state* $q \in states_c$, *it may increment* c *by one and immediately change state, to* $loop_c$ *if* $c < limit_c$ *and to* $continue_c$ *if* $c \geqslant limit_c$. *The automaton terminates as usual when there is no more input, and accepts if there is some sequence of choices that makes it reach a final state.*

Lemma 1. *The problem of determining if a regular expression with exponentiation is* universal, *i.e. that it denotes all strings over its alphabet, is EXPSPACE-complete.*

Proof. See [7].

Lemma 2. *For any regular expression with exponentiation* r, *there exists a NFAC accepting the language* $L(r)$ *that has a number of states that is linear in the number of operators in* r, *and such that* $\Sigma_{c \in C} limit_c \leqslant 2^{|r|}$.

Proof. By structural induction on r. A similar proposition for regular expressions without exponentiation and normal NFA can be found in e.g. [7] and we refer to that for the cases concerning the normal operators.

For $r = r_0^n$, there exists a NFAC $M_0 = \langle Q, \Sigma, \delta, q_0, F, C \rangle$ accepting $L(r_0)$, by the induction assumption. Construct

$$M = \langle Q \cup \{f\}, \Sigma, \delta, q_0, \{f\}, C \cup \{c\} \rangle$$

by setting $states_c = F$, $limit_c = n$, $loop_c = q_0$ and $continue_c = f$.

Any string in $L(r)$ will consist of n substrings such that each is in $L(r_0)$. M starts in state q_0, and since M_0 accepts $L(r_0)$ ends in some state $f_0 \in F$ only after having read one copy of r_0. When, and only when, M enters some $f_0 \in F$ can c be incremented, and therefore M will reach its final state f only after having read n strings in $L(r_0)$. Any string s accepted by M will consist of n consecutive substrings accepted by M_0 and therefore in $L(r_0)$, so $s \in L(r)$.

Since the construction adds a constant number of states for each operator in r, the total number of states is bounded by some linear function in the size of r. Since the total length of all exponents in r

written in binary can not be greater than the length of r, they can not combined represent a greater number than $2^{|r|}$.

Theorem 1. *Deciding existence of a non-branching plan for a scenario $\mathcal{P}$ with propositional domain and containing actions, is EXPSPACE-hard.*

Proof. By reduction from the universality problem for regular expressions with exponentiation, via the same problem for NFACs. For a regular expression with exponentiation, r, we know there exists an NFAC, M, accepting the same language and bounded in $|r|$. We construct a scenario $\mathcal{P}$, polynomially bounded in $|r|$ and such that there exists a plan for $\mathcal{P}$ iff there exists a string in Σ^* not accepted by M. This shows plan existence to be co-EXPSPACE-hard, which since EXPSPACE = co-EXPSPACE, proves the theorem.

We model the state of M and its counters as the state of the scenario, and the transition function δ as the behavior of actions, with one action for each symbol in Σ. Assuming states in Q are numbered $0, \ldots, |Q|$, propositions $s_0, \ldots, s_{\log|Q|}$ denote the current state in binary. Each counter c is likewise represented by propositions $c_0, \ldots, c_{\log limit_c}$. We write $s = m$ for $\bigwedge_{i=0,\ldots,\log|Q|} b_i$, where b_i is s_i if the ith bit of m is 1 and $\neg s_i$ if the ith bit of m is 0, and $s{:=}m$ analogously. For counters c, we write $c = m$ and $c < m$ in the same way, and $c{:=}c+1$ for the conditional reassignment

$$(\neg c_0 \rightarrow c_0{:=}\mathcal{T}) \wedge (\neg c_1 \wedge c_0 \rightarrow c_1{:=}\mathcal{T} \wedge c_0{:=}\mathcal{F}) \wedge \ldots \wedge \\ (\neg c_{\log n} \wedge c_{(\log n)-1} \wedge \ldots \wedge c_0 \rightarrow c_{\log n}{:=}\mathcal{T}; c_{(\log n)-1}{:=}\mathcal{F}; \ldots ; c_0{:=}\mathcal{F})$$

which increments the counter by one, and which is linear in the length of the binary representation of $limit_c$. For each character $\tau \in \Sigma$, define the action

$$A_\tau = \bigwedge (s = i) \rightarrow (\bigvee_{q_j \in \delta(\tau, q_i)} \begin{cases} (s{:=}j) \vee INC(c) & \text{if } q_j \in state_c \\ s{:=}j & \text{if } q_j \notin state_c \end{cases})$$

where $INC(c)$ is the action of incrementing c and chaning state accordingly, encoded as

$$(c < limit_c - 1 \rightarrow c{:=}c+1; s{:=}loop_c) \wedge \\ (c = limit_c - 1 \rightarrow c{:=}c+1; s{:=}continue_c)$$

The initial state I is $s = 0 \wedge (\bigwedge_{c \in C} c = 0)$, which is the initial state of M, and the goal G is $\bigvee_{q_i \notin F} s = i$, the disjunction of all non-accepting states of M.

The construction yields a one-to-one correspondence between plans for $\mathcal{P}$ and sequences of characters that are guaranteed to take M from its initial state to halt in a non-accepting state. Since M accepts exactly the strings that are in $L(r)$, r is universal iff no such plan exists.

The initial state and goal descriptions are bounded by $\log(|Q|) + \Sigma_{c \in C} \log(limit_c)$ and $\log(|Q|) \cdot |F|$, respectively. The number of actions is bounded by $|\Sigma|$, which is at most linear in the size of r, and each action will have no more than $|Q|$ branches. The effect formulas of each branch are bounded by

$$\log(limit_{c_1}) + \ldots + \log(limit_{c_k})$$

which is also linear in the size of r. Therefore, the size of the entire planning scenario, is bounded by $k \cdot r^3$, for some constant k.

To show that plan existence for a propositional scenario with actions is in EXPSPACE, we need a bound on plan length.

Lemma 3. *Let $\mathcal{P} = (\mathcal{D}, \mathcal{O}, I, G)$ be a planning scenario, with $\mathcal{D}$ propositional and $|\mathcal{D}| = n$. If there exists a plan for $\mathcal{P}$, there exists a plan consisting of no more than 2^{2^n} steps.*

Proof. Consider the space of *state sets*, of which there are 2^{2^n}. Since the successor state set function maps a set of states to another, any action is a deterministic transition in this space. Therefore, if there exists a plan for $\mathcal{P}$ there exists also a plan without loops in this space.

Plan existence for a scenario $\mathcal{P}$ with propositional domain and actions can therefore be decided by nondeterministically selecting operators and computing the resulting state set, until the goal is achieved or the length of the sequence exceeds 2^{2^n}. The state set can be represented in space $n \cdot 2^n$ and the counter requires 2^n. Combining this with Theorem 1, we have the following.

Theorem 2. *The non-branching plan existence problem for a scenario $\mathcal{P}$ with propositional domain and actions is EXPSPACE--complete.*

4 Bounded-Length Plans

In response to the hardness results on the complete information planning problem, some researchers have investigated the problem of finding bounded-length plans, mainly plans of polynomial length. It turns out that in a propositional domain, the problem of deciding if there for a (complete information) planning scenario $\mathcal{P}$ exists a plan no longer than $p(|\mathcal{P}|)$, where $p(x)$ is a polynomial, is NP-complete.Thus, some of the hardness of the general planning problem derives from the fact that shortest plans may be very long.

Proposition 1. *Let $\mathcal{P}$ be a planning scenario with propositional domain and actions, and let $p(x)$ be a polynomial. The problem of deciding if there exists a non-branching plan for $\mathcal{P}$ of length at most $p(|\mathcal{P}|)$ is solvable in PSPACE.*

The proof relies on the fact that we can write down a (nondeterministically chosen) candidate plan, then explore every possible execution of this plan depth-first. Since the depth is bounded by the length of the plan, this exploration requires no more than polynomial space (though in the worst case on the order of $p(|\mathcal{P}|)^2$).

5 Limited Information Incompleteness and Dependency

Another way to reduce the complexity of problem is to limit the information incompleteness and/or the context dependency of operators. In this way, we find two severely restricted classes for which the non-branching plan existence problem is PSPACE-complete[3].

Proposition 2. *Let $\mathcal{P}$ be a planning scenario with propositional domain, only deterministic and context independent operators[4] and an arbitrarily incomplete initial state. Then, the non-branching plan existence problem for $\mathcal{P}$ is PSPACE-complete.*

For the second class, we need some definitions.

[3] But still more general than the class of complete information scenarios, for which the plan existence problem is also PSPACE-complete.

[4] That is, essentially STRIPS operators, though with arbitrary preconditions.

Definition 5. *Let $\mathcal{D} = \{p_1, \ldots, p_n\}$ be a propositional domain and let Ξ be a state set over $\mathcal{D}$. We construct a $\mathcal{T}$-$\mathcal{F}$-$\mathcal{U}$* description *of Ξ, $\delta(\Xi)$, by mapping each p_i to $\mathcal{T}$ iff $\xi(p_i) = \mathcal{T}$ for all $\xi \in \Xi$, to $\mathcal{F}$ iff $\xi(p_i) = \mathcal{F}$ for all $\xi \in \Xi$ and to $\mathcal{U}$ ("unknown") otherwise. For a given $\mathcal{T}$-$\mathcal{F}$-$\mathcal{U}$ description, δ, we define the* realisation of δ, *$\rho(\delta)$, as the state set*

$$\{\xi \mid \begin{cases} \xi(p_i) = \mathcal{T} & \textit{if } \delta(p_i) = \mathcal{T} \\ \xi(p_i) = \mathcal{F} & \textit{if } \delta(p_i) = \mathcal{F} \end{cases} \textit{for each } p_i \in \mathcal{D}\}$$

We say a state set Ξ is a $\mathcal{T}$-$\mathcal{F}$-$\mathcal{U}$ state *if $\rho(\delta(\Xi)) = \Xi$, and that an operator O* preserves $\mathcal{T}$-$\mathcal{F}$-$\mathcal{U}$ states *iff $S(O, \Xi)$ is a $\mathcal{T}$-$\mathcal{F}$-$\mathcal{U}$ state whenever Ξ is a $\mathcal{T}$-$\mathcal{F}$-$\mathcal{U}$ state.*

What the $\mathcal{T}$-$\mathcal{F}$-$\mathcal{U}$ property means is that each proposition in the domain is either known or independent of all others. It is similar to the notion of *0-approximation* in [1].

Proposition 3. *Let $\mathcal{P}$ be a planning scenario with propositional domain, an initial state that is $\mathcal{T}$-$\mathcal{F}$-$\mathcal{U}$ and operators that are all $\mathcal{T}$-$\mathcal{F}$-$\mathcal{U}$ preserving. Then, the non-branching plan existence problem for $\mathcal{P}$ is PSPACE-complete.*

The proofs of propositions 2 and 3 both depend on two facts; (*i*) that we can represent the combined effect of a sequence of operators polynomially, without needing to store the actual sequence, and (*ii*) that the incompleteness is constant or decreasing, which limits the length of the shortest plan to single exponential.

Definition 5 is rather technical, and a more intuitive characterization of $\mathcal{T}$-$\mathcal{F}$-$\mathcal{U}$ preserving operators is difficult to find. Deterministic context independent operators preserve $\mathcal{T}$-$\mathcal{F}$-$\mathcal{U}$ states, as do certain kinds of deterministic context dependent operators[5]. Nondeterministic operators that are purely information-destructive, *i.e.* that assign to the nondeterministically affected propositions any possible combination of values, also preserve $\mathcal{T}$-$\mathcal{F}$-$\mathcal{U}$ states.

[5] Note, however, that this does not imply that proposition 3 subsumes proposition 2, since the later allows arbitrary initial state.

6 Partially Observable Domains and Branching Plans

In the presence of incomplete information, or even nondeterminism, it may seem as a good idea to equip the executing agent with some form of sensors and to conditionalize plan execution on the results of sensing; that is, to search for branching plans.

In the case of Markov decision processes, the policy existence problems for both fully observable and unobservable domains are known to be somewhat easier than the corresponding problem for partially observable domains. By analogy, it is not unreasonable to expect the branching plan existence problem for partially observable domains with nondeterministic context dependent operators to be at least as hard as the problem corresponding non-branching plan existence problem, if not harder.

However, we can encode a limited form of "branchingness" in a non-branching plan using operator context dependency. Specifically, for a scenario $\mathcal{P}$ with propositional domain of size n and actions, by adding 2 domain propositions and $2n+3$ actions (depending only on the domain)[6], allows the construction of plans of the form

$$\textbf{IF } \gamma_1 \textbf{ THEN } A_1;\ \textbf{IF } \gamma_2 \textbf{ THEN } A_2; \ldots;\ \textbf{IF } \gamma_m \textbf{ THEN } A_m$$

where each γ_i is an arbitrary formula over the domain propositions, The idea is to write each γ_i on disjunctive normal form and evaluate it "linearly", storing the intermediate results in two control propositions.

A partially observable domain can be encoded in an unobservable domain, by duplicating propositions, letting one set represent the actual world state and the other the agents knowledge of the state. Normal operators would then depend on and effect only the "world" propositions, while sensing operators change the "knowledge" propositions depending on the corresponding world propositions.

Because both transformations are polynomial, this means that at least a limited form of the contingent plan existence problem in partially observable domains is no harder to decide than conformant plan existence in an unobservable domain.

[6] A constant change to each action originally in $\mathcal{P}$ is also necessary.

7 Conclusions and Future Work

The high complexity of the planning problem in the presence of nondeterminism and context dependency should come as no surprise. This result is however interesting to compare with the corresponding problem for probabilistic domains, known to be NEXPTIME-complete and thus presumably easier[7]. That nondeterministic choice contains less information than probabilistic choice is a known fact, but these results together show that for the planning problem, the difference is significant.

This paper also leaves a large number of open questions. Is the polynomially bounded plan existence problem discussed in section 4 hard for PSPACE? To show this by reduction meets with some difficulties, since known PSPACE-complete problems involve either exponentially long sequences (for instance, the reachable deadlock problem [11]) or choice (such as the satisfiability problem for quantified Boolean formulas and various two-player games). If this problem is not hard for PSPACE, is there a more efficient algorithm that solves it?

What is the complexity of the (fully) branching plan existence problem on fully or partially observable domains? The encoding shown in section 6 depends only on the problem domain, and is linear in size. If it is also allowed to depend on operators, and increase in size proportional to a higher degree polynomial, are there then other classes of branching plans that can be encoded?

Acknowledgments

This research has been sponsored by the *Swedish Research Council for the Engineering Sciences* (TFR) under grant Dnr. 97-301, the *Wallenberg Foundation* and the *ECSEL/ENSYM* graduate studies program.

[7] There is a class called $\bigcup_{k>0} TA[2^{n^k}, n]$ that lies inbetween NEXPTIME and EXPSPACE, and is believed to be strictly inbetween. This fascinating class captures the decision problem for the theory of the reals with addition. For details, see [8].

References

1. C. Baral, V. Kreinovich, and R. Trejo. Computational complexity of planning and approximate planning in presence of incompleteness. In *Proc. 16th International Joint Conference on Artificial Intelligence (IJCAI'99)*, 1999.
2. T. Bylander. Complexity results for planning. In *Proceedings of the 12th International Joint Conference on Artificial Intelligence (IJCAI'91)*, 1991.
3. G. Collins and L. Pryor. Planning under uncertanity: Some key issues. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI'95)*, 1995.
4. D. Draper, S. Hanks, and D. Weld. Probabilistic planning with information gathering and contingent execution. In *Artificial Intelligence Planning Systems: Proceedings of the 2nd International Conference (AIPS'94)*, 1994.
5. K. Erol, D.S. Nau, and V.S. Subrahmanian. Complexity, decidability and undecidability results for domain-independent planning: A detailed analysis. Technical Report CS-TR-2797, Computer Science Department, University of Maryland, 1991.
6. J. Goldsmith, C. Lusena, and M. Mundhenk. The complexity of deterministically observable finite-horizon markov decision processes. Technical Report 268-96, Computer Science Department, University of Kentucky, 1996.
7. J.E. Hopcroft and J.D. Ullman. *Introduction to Automata theory, Languages and Computation.* Addison-Wesley, Reading, MA, 1979.
8. D.S. Johnson. A catalog of complexity classes. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume A. Elsevier, Amsterdam, 1990.
9. N. Kushmerick, S. Hanks, and D. Weld. An algorithm for probabilistic least-commitment planning. In *Proceedings of the 12th National Conference on Artificial Intelligence (AAAI'94)*, 1994.
10. M.L. Littman. Probabilistic propositional planning: Representations and complexity. In *Proceedings of the 14th National Conference on Artificial Intelligence (AAAI'97)*, 1997.
11. C.H. Papadimitrou. *Computational Complexity.* Addison-Wesley, Reading, MA, 1994.
12. M. Peot and D. Smith. Conditional nonlinear planning. In *Artificial Intelligence Planning Systems: proceedings of the International Confrence (AIPS'92)*, 1992.
13. D.H.D. Warren. Generating conditional plans and programs. In *Proceedings of the Summer Conference on AI and Simulation of Behaviour*, 1976.
14. D. Weld, C. Anderson, and D. Smith. Extending Graphplan to handle uncertainty & sensing actions. In *Proc. 15th National Conference on Artifical Intelligence (AAAI'98)*, 1998.
15. D. Weld and D. Smith. Conformant Graphplan. In *Proc. 15th National Conference on Artifical Intelligence (AAAI'98)*, 1998.