Probabilistic Propositional Planning: Representations and Complexity

Michael L. Littman
Department of Computer Science
Duke University, Durham, NC 27708-0129
mlittman@cs.duke.edu

Abstract

Many representations for probabilistic propositional planning problems have been studied. This paper reviews several such representations and shows that, in spite of superficial differences between the representations, they are "expressively equivalent," meaning that planning problems specified in one representation can be converted to equivalent planning problems in any of the other representations with at most a polynomial factor increase in the size of the resulting representation and the number of steps needed to reach the goal with sufficient probability. The paper proves that the computational complexity of determining whether a successful plan exists for planning problems expressed in any of these representations is EXPTIME-complete and PSPACE-complete when plans are restricted to take a polynomial number of steps.

Introduction

In recent years, there has been an interest in solving planning problems that contain some degree of uncertainty. One form that this uncertainty has taken is uncertainty in the effect of executing an action: because of unmodeled or uncontrollable influences, different effects are possible as a result of executing a given action in a given state. In *probabilistic planning* (Kushmerick, Hanks, & Weld 1995), these uncertainties are modeled as probabilities and the planning problem becomes that of determining whether a plan exists that succeeds with sufficient probability.

Although a great deal is known about the role of representation in and computational complexity of deterministic propositional planning (Bäckström 1995; Bylander 1994), probabilistic propositional planning has received much less attention. Several recent planners have been designed to manipulate probabilistic representations (Kushmerick, Hanks, & Weld 1995; Draper, Hanks, & Weld 1993; Dearden & Boutilier 1997; Boutilier, Dearden, & Goldszmidt 1995), which express a planning problem as a compact or factored Markov decision process (MDP).

This paper begins to explore the role of representation in probabilistic planning. The first section introduces one representation, ST (sequential-effect trees), for probabilistic propositional planning problems. The next section shows that determining whether a plan exists that can achieve a goal state with sufficient probability in an ST planning problem is EXPTIME-complete. The following section examines a simpler problem in which the space of plans is limited to those that take a polynomial number of steps; in this case, the plan-existence problem is PSPACE-complete. The next section examines ST and a number of other common representations for expressing probabilistic propositional planning problems and proves that they are all equivalently expressive in terms of how compactly they can formulate different problems; this implies that the complexity results for ST apply to many alternate representations as well.

Basic Definitions

For the purposes of this paper, a propositional probabilistic planning problem is defined by a finite set of n distinct *propositions*, any of which may be true or false at any (discrete) time t. A subset of these propositions constitute the *initial state* s_0, which is the set of propositions that are true at $t = 0$. Another subset of the propositions is the *goal set* g, and any state $s \supseteq g$ is considered to be a goal state.

Each of a set of *actions* transforms the set of true propositions at time t into a set of true propositions at time $t + 1$. Each action **a** induces a probability distribution over the power set of propositions (that is, a probability distribution over states).

In the sequential-effects tree representation (ST), the effect of each action on each proposition is represented as a separate decision tree. A given action **a** can change some ordered subset of propositions p_1 through p_k. The decision tree for proposition p_i can refer to the values of propositions at time t and the values of propositions p_1 through p_{i-1} at time $t + 1$. The leaves of a decision tree describe how the associated proposition

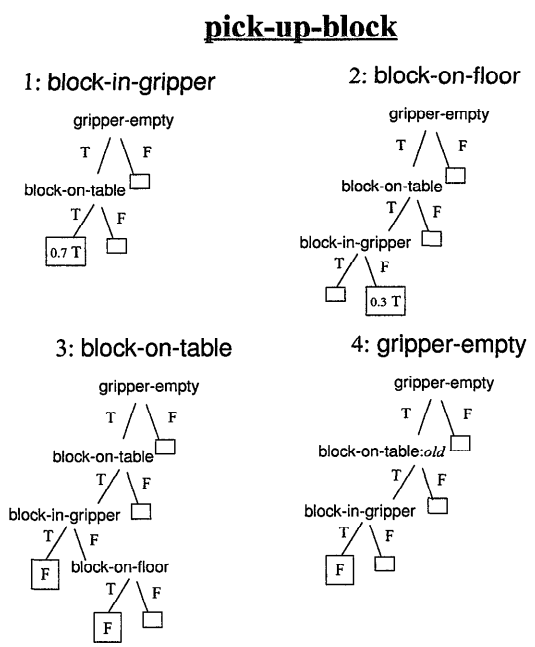

Figure 1: ST representation for an example action.

changes as a function of the state and action.

Figure 1 illustrates the ST representation of a simple probabilistic action that might be part of a stochastic blocks world. This **pick-up-block** action has three possible outcomes: success (block is picked up), failure (no change), or unintended consequence (block is knocked to the floor). The probabilities of these three outcomes are 0.7, 0.21, and 0.09, respectively. The first decision tree determines whether or not the block-in-gripper proposition will be true after the action is executed. The decision tree has two internal nodes labeled with *decision propositions* corresponding to the two precondition propositions for this action. If both of these decision propositions are true, then the block is picked up into the gripper successfully with probability 0.7. The second decision tree determines whether the block is knocked to the ground. If the block is not in the gripper after the first decision tree is evaluated, the block ends up on the floor with probability $0.09/(1-0.7) = 0.3$. Note that an empty box at a leaf means no change, "T" means the proposition becomes true, "F" means it becomes false, and "p T" means it becomes true with probability p and false with probability $1-p$.

The remaining two decision trees adjust the block-on-table and gripper-empty propositions as a function of the new values of block-in-gripper and block-on-floor. Note that the fourth decision tree needs to make use of the old value of block-on-table, as opposed to the value defined in the third decision tree; this is accomplished by adding the suffix "*old*" to the appropriate decision proposition.

Given a propositional probabilistic planning problem and some probability value θ, the *plan-existence* problem is to determine whether there is any way to select an action to apply at each step t (perhaps as a function of the current state s_t) such that a goal state is eventually reached with probability at least θ. In the *polynomial-horizon plan-existence* problem, we are only interested in knowing whether the goal can be reached with probability at least θ before some time limit l, where l is bounded by a polynomial in the size of the planning problem.

For problems in ST, the plan-existence problem is EXPTIME-complete and the polynomial-horizon plan-existence problem is PSPACE-complete. We prove that many other representations are "expressively equivalent" (Bäckström 1995) to ST, and therefore the same complexity results apply to a number of important representations.

Plan Existence

This section shows that the plan-existence problem for probabilistic propositional planning problems in ST is EXPTIME-complete. Since EXPTIME is not as well known as either NP or PSPACE, it deserves some further explanation. Additional background on complexity classes can be found in Papadimitriou's textbook (1994).

The class NP is the set of problems that can be solved in nondeterministic polynomial time. NP-complete problems are the hardest problems in NP; if they could be solved in deterministic polynomial time (P), then all NP problems could. Many interesting and natural problems are NP-complete and the best algorithms known for these problems take exponential time. PSPACE is the set of problems solvable in polynomial space. It contains NP and P, although it is not known whether the containment is proper—it is possible that PSPACE = NP = P and all problems in PSPACE can be solved in polynomial time. PSPACE-complete problems are the hardest problems in PSPACE and the best algorithms known for these problems take exponential time.

The class EXPTIME (or EXP) is the set of problems that can be solved in exponential time. PSPACE, NP, and P are all contained within EXPTIME. Although it is not known whether NP = PSPACE = EXPTIME, we do know that P $\neq$ EXPTIME (Papadimitriou 1994). This means that there are problems in EXPTIME that *cannot* be solved in polynomial time. The EXPTIME-complete problems are the hardest problems in EXPTIME, so proving that a problem is EXPTIME-complete shows that it is not solvable in polynomial time in the worst case.

There are some strong reasons to suspect that the probabilistic propositional plan-existence problem is EXPTIME-complete. First, it seems harder than deterministic plan existence, which is PSPACE-complete (Bylander 1994). Second, it is a compact

form of the problem of solving an MDP. Solving an MDP is P-complete (Papadimitriou & Tsitsiklis 1987) and it is often the case that succinctly represented versions of a problem are exponentially harder to solve than their simple versions (Papadimitriou 1994); this would imply EXPTIME-completeness if it were true in general.

To prove that these intuitions are correct, we need to show that the plan-existence problem is no easier than some EXPTIME-complete problem; we use a two-player game, played on 13-DNF boolean formulae, called G_4 (or "Peek"). This somewhat peculiar game was devised by Stockmeyer and Chandra (1979) in their paper linking combinatorial two-player games to the class EXPTIME; it is one of the few canonical EXPTIME-complete problems.

The game is played as follows. The "board" is a 13-DNF formula with a set of assignments to its $2k$ boolean variables. One set of variables $x_1 \ldots x_k$ belongs to player 1 and the rest $y_1 \ldots y_k$ to player 2. Players take turns flipping the assignment of one of their variables. The game is over when the 13-DNF formula evaluates to true with the winner being the player whose move caused this to happen. The computational problem is to determine whether there is a winning strategy for player 2 for a given formula from a given initial assignment of the variables.

Solving such a game can be reduced to the problem of plan existence in a probabilistic propositional domain. The role of player 1 is played by the decision maker (the planner) and the role of player 2 is played by "chance" (stochastic transitions). If we associate a winning state for player 1 with a goal state in the corresponding planning problem, a winning strategy for player 1 corresponds to a plan that reaches a goal state with probability 1.

Theorem 1 *The plan-existence problem for ST is EXPTIME-complete.*

Proof: To show EXPTIME-completeness, we show that the problem can be solved in exponential time and that the EXPTIME-hard problem G_4 can be reduced to it.

To solve the plan-existence problem in exponential time, simply create an MDP with one state for each possible subset of propositions. The number of states in the MDP is 2^n and the number of actions is precisely the number of actions in the planning problem. Such an MDP can be solved in exponential time (polynomial in the number of states) using linear programming (Condon 1992). If the optimal value of the initial state s_0 exceeds the threshold θ, a valid plan exists.

Next, we show how an instance of the G_4 game can be represented in ST. The state of the game is encoded as propositions and the different choices that can be made in the course of a game are encoded as actions; Figure 2 illustrates the actions that define the moves of the game.

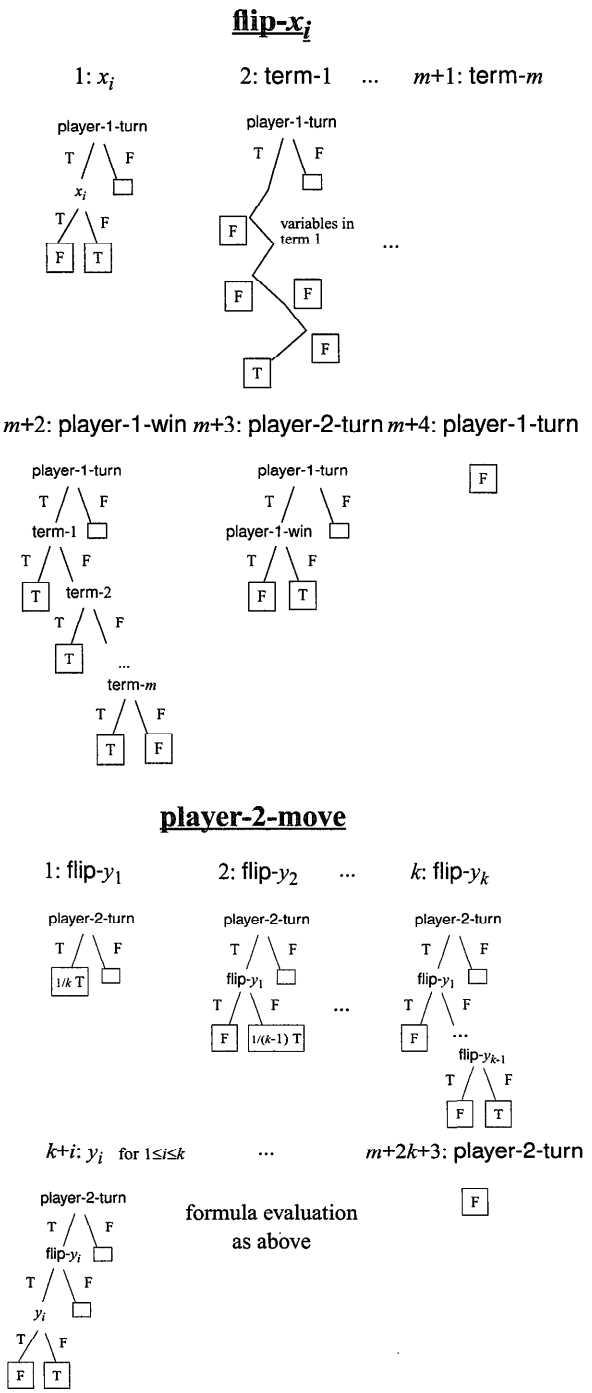

Figure 2: ST representation for actions in G_4.

The flow of the game is controlled by two propositions: player-1-turn and player-2-turn. The current assignment is encoded by a set of propositions: $x_1 \ldots x_k$, for player 1 and $y_1 \ldots y_k$ for player 2. When it is player 1's turn, player 1 can choose to flip the assignment of one of its variables. The **flip-x_i** actions, for example, only have an effect when player-1-turn is true. For a given value of i, **flip-x_i** deterministically makes x_i take on the opposite value, then if one of the terms of the DNF formula has become true, player-1-win becomes true, otherwise, player-2-turn becomes true and player-1-turn false, turning control back to player 2.

The action that defines player 2's moves, **player-2-move**, is more complex. It randomly chooses a variable to flip (caching the decision in the flip-y_i propositions), then flips the variable, then checks to see whether this results in a win for player 2. If it does not, control is returned to player 1. We needn't assume that player 2 chooses actions "intelligently"; any non-zero probability that player 2 makes a good sequence of choices is sufficient for player 2 to win some of the time.

The initial state is { player-1-turn } unioned with the set of propositions that specify the initial assignment. The goal set is { player-1-win }. If there are k variables for player 1, k variables for player 2, and m terms in the DNF formula, then the total number of propositions in the problem is $3k+m+4$ and the total number of actions is $k+1$. The sizes of the ST representations of the various actions are polynomial in the size of the G_4 instance. The resulting propositional planning problem can be satisfied with probability 1 if and only if the given instance of G_4 does not have a guaranteed win for player 2. This is because if there is a guaranteed win for player 2, there is a guaranteed win in a finite number ($2^{O(k)}$) of steps. Therefore, a randomized strategy for player 2 will win with non-zero probability against an optimal player 1 if there is a guaranteed win for player 2. If there is no guaranteed win for player 2, then the game is either a win for player 1 or a draw. When playing against a randomized strategy for player 2, a draw is a zero-probability event—player 2's variables take on every possible assignment with probability 1, so player 1 is guaranteed to be able to win eventually.

Since determining whether an instance of G_4 has a guaranteed win for player 1 is EXPTIME-hard, solving the corresponding plan-existence problem is EXPTIME-hard. □

Polynomial-Length Plans

In this section, we consider a restriction of the general plan-existence problem to only "short" plans; the resulting problem is PSPACE-complete. Although this is the same complexity class as the plan-existence problem in deterministic domains (Bylander 1994), the two problems are difficult for very different reasons. In the case of deterministic planning, the problem is closely related to that of determining reachability in an exponential-sized graph; polynomial-horizon probabilistic planning, on the other hand, is more like searching a polynomial depth AND-OR tree, and the difficulty comes, not from the exponential length of paths (they are all of polynomial length), but instead from the exponential number of leaves in the tree. This analysis suggests a relationship between this planning problem and the quantified boolean formula problem, and it is this relationship that is exploited in the PSPACE-completeness proof.

The quantified boolean formula problem (QBF) is specified by a 3-CNF formula Ψ on a set of $2k$ variables, $x_1 \ldots x_k, y_1 \ldots y_k$. The problem is to determine whether the formula

$$\exists x_1 \forall y_1 \exists x_2 \forall \ldots \exists x_k \forall y_k \Psi \qquad (1)$$

is true.

Solving the QBF problem can be reduced to the problem of plan existence in a polynomial-horizon probabilistic propositional domain. The bindings for the x_i variables are chosen by the planner, and the bindings for the y_i variables are chosen at random. If the goal can be reached with probability 1, then Equation 1 is true. This formulation of QBF has also been studied in the "games against nature" model (Papadimitriou 1983).

Theorem 2 *The polynomial-horizon plan-existence problem for ST is PSPACE-complete.*

Proof: To show PSPACE-completeness, we show that the problem can be solved in polynomial space and that the PSPACE-hard problem QBF can be reduced to it.

We can solve the l-horizon plan-existence problem in polynomial space by evaluating a depth $2l$ tree of states rooted at the initial state in a depth-first fashion. The value of a leaf is 1 if it is a goal state and 0 otherwise. The levels of the tree alternate between action levels and chance levels. At action levels, the branching factor is equal to the number of actions and the value is the maximum value of any of the children. At chance levels, the branching factor is equal to the number of states reachable with non-zero probability when the most recent action is chosen from the most recent state. The value of a chance node is the probability weighted average of the children nodes. The value of the root node becomes the maximum probability of reaching a goal state in l steps from the initial state; if this value exceeds the threshold θ, a valid plan exists. The amount of space needed to compute this value is a polynomial amount of space for each node along a length $2l$ path from the root to a leaf. Since we are assuming l is polynomially bounded, the total space requirements are polynomial. This proves membership in PSPACE.

To prove PSPACE-hardness, we show how an instance of QBF can be represented in ST. The different choices that can be made in the course of evaluating

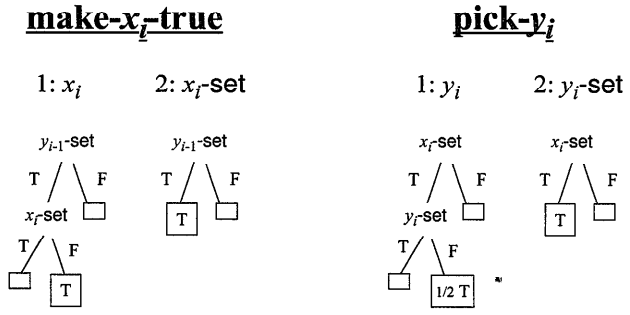

Figure 3: ST representation for two QBF actions.

the formula are encoded as ST actions; Figure 3 illustrates several of these actions.

The propositions x_i-set and y_i-set control the sequence of events. A successful plan alternates between choosing whether to make one of the x variables true and randomly choosing a value for a y variable, until all $2k$ variables have been assigned values. Then an **evaluate-formula** action is chosen which results in the proposition formula-true if and only if the formula is true.

The initial state is { y_0-set } and the goal set is { formula-true }. If there are m clauses in the CNF formula, then the total number of propositions in the problem is $4k + m + 2$ and the total number of actions is $3k + 1$. The sizes of the ST representations of the various actions are polynomial in the size of the QBF instance. The resulting propositional planning problem can be satisfied with probability 1 if and only if the given instance of QBF is true. Since determining whether an instance of QBF is true is PSPACE-hard, solving the corresponding plan-existence problem is PSPACE-hard. □

A similar result was proven recently by Goldsmith, Lusena and Mundhenk (1996), where the actions were represented by succinct circuits.

Representational Equivalence

In this section, we consider several related representations for probabilistic planning problems. On the surface, these representations appear to vary significantly in their expressiveness; it seems that some of the representations ought to be able to express certain MDPs much more succinctly than the others. We show, in fact, that this is not the case; all of the representations are expressively equivalent.

We consider a fairly strong type of equivalence, in which two representations are equivalent if and only if a planning problem expressed in one representation can be transformed into the other with at most a polynomial increase in the number of propositions and actions and at most a polynomial increase in the size of the action descriptions. In addition, every sequence of actions that reaches a goal in the first representation must be transformable to a sequence of actions that reaches the goal in the second representation without introducing any new paths to the goal, altering the probability of reaching the goal at all, or increasing the length of the sequence by more than a polynomial factor.

This type of equivalence has both practical and theoretical significance. On the theoretical side, a problem that is NP- or PSPACE- or EXPTIME-complete in one representation will be the same in any equivalent representation. This means that, for all the representations described here, the plan-existence problem is EXPTIME-complete and the polynomial-horizon plan-existence problem is PSPACE-complete. On the practical side, planning algorithms designed for planning problems in one representation can be used to solve planning problems expressed in any equivalent representation by a simple transformation. The equivalence proofs are based on "exact structure preserving" (ESP) reductions, proposed by Bäckström (1995) in the context of deterministic planning.

The probabilistic planning representations considered in this section are closely related to the sequential-effect trees representation (ST) described earlier. In ST, the effects of an action on the state are described by a set of decision trees, one for each proposition that can change, that give the conditions under which the propositions change. We consider two primary dimensions along which representations can vary. The conditions under which the propositions change can be represented by full boolean tables (F), or decision trees (T), and the propositions can be defined sequentially (S) or independently (I). In the independent representation, the conditions under which a proposition changes can only be expressed as a function of the value of the propositions in the previous state; there can be no sequential dependencies. Combining these attributes leads to four representations: ST, SF, IF, and IT.

Boutilier, Dean, and Hanks (1995) call the graphical form of these representations "two-stage temporal Bayes networks" (2TBNs); the basic 2TBN representation corresponds to SF[2]. They use the term "simple 2TBN" for IF, and mention ST and IT as a way of exploiting "propositional independence."

In their discussion of the advantages and disadvantages of the different representations, Boutilier, Dean and Hanks suggest that the representations are somehow fundamentally different; planners that are efficient for one representation might not be efficient for domains represented in another. They also argue that IF and IT are less powerful and that domains that can be represented succinctly in ST and SF become exponentially larger when expressed in IF and IT. This

[2]Propositions correspond to variables, boolean tables to CPTs, and the ordering of condition tables to a topological sort of the 2TBN.

section proves that, to the contrary, all four representations are equivalent (to within a polynomial factor); any planning problem specified compactly in any of the representations can be represented compactly in all of them. From a worst-case complexity standpoint, all of these representations are the same.

Boutilier et al. also mention other representations, including acyclic decision graphs for representing conditions and "factored probabilistic state-space operators" (PSOs) for representing correlated effects. These representations can also be proven equivalent to ST and the others, although there is not sufficient space to present this here.

Theorem 3 *The following representations are expressively equivalent: IF, IT, SF, and ST.*

Proof: Since independent effects are a special case of sequential effects and a full table can be represented by a complete binary decision tree, it is clear that IF can be transformed to any of IT, SF, or ST with at most a constant factor increase in the representation size. Similarly, IF, IT, and SF can all be transformed into ST with at most a constant factor increase in the representation size.

To complete the proof, we next argue that planning problems in ST can be transformed into planning problems in IF increasing the representation size and the length of paths to the goal by at most a polynomial factor.

The basic idea behind the transformation is that we can trade complexity in the representation of conditions for complexity in the paths to the goal by using a sequence of IF actions to simulate each ST action.

1. Make a copy of each proposition, so that we can talk about propositions in the previous state and propositions in the new state. For each proposition p, this creates a proposition p:*old*, which represents the value that p had in the previous state *after* proposition p takes on its new value. This process doubles the number of propositions in the planning problem.
2. Add a new proposition called no-error and one called action-in-progress. The proposition no-error is in the initial state and in the goal set. Once it becomes false, no action will make it true again and therefore the goal will become unreachable. The action-in-progress proposition is initially false and is false each time a decision must be made; it is true while an ST action is being simulated using a sequence of IF actions. This increases the set of propositions by two.
3. For each node n in each of the ST decision trees representing the actions, add a new action **n** and a new proposition n. The new propositions are all initially false. The representation for action **a** becomes a simple action that makes no-error false if action-in-progress is true, and otherwise makes action-in-progress and the proposition associated with the first node of the ST decision tree representing action **a** true. This increases the number of propositions and the number of actions by a factor equal to the size of the ST representation for the planning problem.
4. Let n be a node in the ST decision tree for action **a** and proposition p. Either n is an internal node or a leaf node. If n is an internal node, let d be the decision proposition at n, ℓ be n's left child and r be n's right child. The action **n** associated with n works as follows. If the proposition n associated with n is false, no-error becomes false. If d is true, it makes n false and l, the proposition associated with l, true. If d is false, it makes n false and r, the proposition associated with r, true.
5. On the other hand, if n is a leaf node labelled with "p T", the action **n** is defined as follows. If n is false, no-error becomes false. Next, p:*old* takes the value that p has, and p becomes true with probability p. Finally, n is made false, and the proposition associated with the root of the next ST decision tree that needs to be processed is made true. If the current ST decision tree is the final one in the representation, then action-in-progress is made false.

All these actions use at most a constant number of propositions in their conditions, therefore they can be represented in IF with the size of each action representation being no more than some constant.

This transformation results in an IF representation of a planning problem that is no more than a polynomial factor larger than the given ST planning problem. In addition, any sequence of states from the initial state to the goal state in the ST version of the problem corresponds to a sequence of states from the initial state to the goal state in the IF version of the problem; the probability of this sequence is the same as the probability of the corresponding sequence in the ST problem and the path is no more than a polynomial factor longer. □

Discussion

This paper shows that the plan-existence problem for probabilistic planning problems in ST, a syntactic variant of two-stage temporal Bayes networks with conditional probability tables represented as decision trees, is EXPTIME-complete and PSPACE-complete when only a polynomial number of actions can be executed to reach the goal.

The paper also shows that several variants of ST are equivalent in a very strong sense. These representational equivalence results are perhaps surprising because the representations involved appear to be quite different in their expressive power. However, by trading complexity in action representation for complexity in plan execution, the representations can be made to express the same problems with nearly the same degree of succinctness. The equivalence results can be

combined with the computational complexity results to show that the computational intractability of the plan-existence problem holds across a wide variety of different representations.

Many extensions to this work are possible. In light of the worst-case difficulty of the plan-existence problem, it is natural to seek out approximation algorithms and algorithms that perform well on average. The plan-existence problem is no easier to approximate to within any constant factor c of the exact threshold θ; this is because the approximate version of the problem is equivalent to using a threshold of θ/c in the exact version of the problem. (The hardness proofs use $\theta = 1$, but these can be adapted to lower values of θ by introducing probabilistic transitions to sink states.)

Another possible direction is to consider restrictions on the representation (e.g., limiting the size or depth of the ST decision trees). This type of analysis has been carried out for deterministic planning (Bylander 1994); it is likely that unrealistically strong restrictions will be necessary to render the probabilistic propositional planning problem tractable.

A more promising approach is to seek out algorithms that perform well in practice. The transformations described in this paper preserve worst-case complexity, but can have a substantial impact on best-case performance. This is because representational size is not all that matters. Just as the use of "noisy-or" nodes can speed up inference in belief nets (Heckerman & Breese 1994), representations and algorithms that exploit additional structure in propositional planning problems can be of great practical value.

It is worth noting that the completeness results given in this paper also hold for the plan-existence problem in non-deterministic domains with only a slight change in the details of the proofs.

The strong negative results reported in this paper must be put into perspective so as not to be interpreted too pessimistically. In particular, the plan-existence problem is not the most significant problem that planning systems need to solve; even if we *could* solve it quickly, it wouldn't do us too much good because the plans themselves might be extremely difficult to write down and could take arbitrarily long to execute. A more practical issue is to determine whether there is a good *succinct* plan for a given planning problem. This is also related to the problem of creating plans in partially observable domains (Draper, Hanks, & Weld 1993) and its complexity has been addressed in recent work (Goldsmith, Littman, & Mundhenk 1997); the most important variation is NP^{PP}-complete and therefore possibly more amenable to approximation than the PSPACE-hard problems described in this paper.

Acknowledgments. Thanks to Tom Bylander, David McAllester, Hagit Shatkay, Dan Roth, Sonia Leach, Craig Boutilier and an anonymous reviewer for feedback and suggestions.

References

Bäckström, C. 1995. Expressive equivalence of planning formalisms. *Artificial Intelligence* 76(1–2):17–34.

Boutilier, C.; Dean, T.; and Hanks, S. 1995. Planning under uncertainty: Structural assumptions and computational leverage. In *Proceedings of the Second European Workshop on Planning.*

Boutilier, C.; Dearden, R.; and Goldszmidt, M. 1995. Exploiting structure in policy construction. In *Proceedings of the International Joint Conference on Artificial Intelligence.*

Bylander, T. 1994. The computational complexity of propositional STRIPS planning. *Artificial Intelligence* 69:161–204.

Condon, A. 1992. The complexity of stochastic games. *Information and Computation* 96(2):203–224.

Dearden, R., and Boutilier, C. 1997. Abstraction and approximate decision-theoretic planning. *Artificial Intelligence* 89(1–2):219–283.

Draper, D.; Hanks, S.; and Weld, D. 1993. Probabilistic planning with information gathering and contingent execution. Technical Report 93-12-04, University of Washington, Seattle, WA.

Goldsmith, J.; Littman, M.; and Mundhenk, M. 1997. The complexity of plan existence and evaluation in probabilistic domains. Technical Report CS-1997-07, Department of Computer Science, Duke University.

Goldsmith, J.; Lusena, C.; and Mundhenk, M. 1996. The complexity of deterministically observable finite-horizon Markov decision processes. Technical Report 268-96, Department of Computer Science, University of Kentucky.

Heckerman, D., and Breese, J. S. 1994. Causal independence for probability assessment and inference using Bayesian networks. Technical Report MSR-TR-94-08, Microsoft Research, Advanced Technology Division.

Kushmerick, N.; Hanks, S.; and Weld, D. S. 1995. An algorithm for probabilistic planning. *Artificial Intelligence* 76(1-2):239–286.

Papadimitriou, C. H., and Tsitsiklis, J. N. 1987. The complexity of Markov decision processes. *Mathematics of Operations Research* 12(3):441–450.

Papadimitriou, C. H. 1983. Games against nature (extended abstract). In *24th Annual Symposium on Foundations of Computer Science,* 446–450.

Papadimitriou, C. H. 1994. *Computational Complexity.* Reading, MA: Addison-Wesley.

Stockmeyer, L. J., and Chandra, A. K. 1979. Provably difficult combinatorial games. *SIAM Journal of Computing* 8(2):151–174.