Polynomial-Length Planning Spans the Polynomial Hierarchy

Hudson Turner

University of Minnesota, Duluth
`hudson@d.umn.edu`

Abstract. This paper presents a family of results on the computational complexity of planning: classical, conformant, and conditional with full or partial observability. Attention is restricted to plans of polynomially-bounded length. For conditional planning, restriction to plans of polynomial size is also considered. For this analysis, a planning domain is described by a transition relation encoded in classical propositional logic. Given the widespread use of satisfiability-based planning methods, this is a rather natural choice. Moreover, this allows us to develop a unified representation—in second-order propositional logic—of the range of planning problems considered. By describing a wide range of results within a single framework, the paper sheds new light on how planning complexity is affected by common assumptions such as nonconcurrency, determinism and polynomial-time decidability of executability of actions.

1 Introduction

This paper presents a family of results on the computational complexity of planning problems in which a plan of polynomially-bounded length is sought. A planning problem traditionally consists of three elements: a description of the planning domain (specifying how actions can affect the state of the world), a description of the initial state of the world, and a description of the goal. A solution to a planning problem is a *valid* plan—one whose execution is guaranteed to achieve the goal, and whose executability is also guaranteed. In investigating the computational complexity of planning, it is convenient to focus on the associated decision problem: Does a valid plan exist?

In classical planning, the initial state is completely specified, the action domain is deterministic, and a plan is a finite sequence of actions to be performed. These assumptions can be relaxed to obtain larger classes of planning problems. For instance, in conformant planning, one no longer assumes that the initial state is completely specified, and actions are allowed to have nondeterministic effects, but a plan is still a sequence of actions. Another class of planning problems is obtained by allowing conditional plans, in which the execution of a plan at each step may depend on the previous execution history. In characterizing this dependence, we may assume that the current state is fully observable or only partially observable during plan execution. The paper considers the complexity of each of these classes of planning problems.

This is not a new line of research; among the papers with results on planning complexity published in the last decade are [3, 1, 6, 10, 11, 4, 8, 14, 2, 5]. The work presented here is unusual in employing a single framework to study the complexity of such a wide range of polynomial-length planning problems.

In fact, one contribution of the paper is a unified representation—in second-order propositional logic—of the decision problems associated with classical, conformant and conditional planning (with full or partial observability). In many cases, membership of a planning problem in a given complexity class can be determined by the form of its QBF representation. Moreover, several of the hardness results are especially easy to obtain in this setting.

The analysis employs a simple, general action representation framework in which a planning domain is described by a transition relation encoded in classical propositional logic. This is technically convenient, given our interest in QBF representations, and also sensible in light of the widespread use of satisfiability methods in automated planning, which depends on the fact that such descriptions can be obtained by polynomial-time translation from many other action description languages. Thus, the results obtained here have wide applicability.

Moreover, this action representation framework allows for such complications as nondeterminism and concurrent execution of actions, which makes it convenient to consider the incremental effect of imposing common planning assumptions such as determinism and nonconcurrency. Consequently, our analysis sheds new light on the role such assumptions play in determining the complexity of each of the families of planning problems considered.

Many accounts of planning are based on action representations with another significant (but less often remarked on) property: the executability of actions in a state can be determined in polynomial time. This assumption does not hold for the action representation framework used here. Let us consider very briefly why it can be convenient to abandon it. In general, this assumption will not hold for action representations that allow for *implied* action preconditions. For example, a domain description may describe the fact that two blocks cannot occupy the same position, and from this general fact many action preconditions may follow. For instance, it can be inferred that block B_1 cannot be moved to the location of block B_2 (unless block B_2 is moved somewhere else). It also follows that blocks B_1 and B_2 cannot be moved concurrently to the same location.

A number of results reported here are similar to (although distinct from) results in [10, 14, 2, 5]. In each case this will be noted when the result is encountered in the text. (A more comprehensive survey of planning complexity results is beyond the scope of this paper.) Due to space restrictions, proofs are omitted.

2 Preliminaries

2.1 Action Representation Framework

Begin with a set A of action symbols and a disjoint set F of fluent symbols. Let $\mathit{state}(F)$ be a formula in which the only nonlogical symbols are elements of F.

The formula $state(F)$ encodes the set of states that correspond to its models. Let $act(F, A, F')$ be a formula with nonlogical symbols taken from $F \cup A \cup F'$, where F' is obtained from F by priming each element of F. (Assume F' is disjoint from $F \cup A$. From now on similar disjointness assumptions will be left unstated.) Let $state(F')$ be the formula obtained from $state(F)$ by substituting for each occurrence of each fluent symbol its primed counterpart. (Throughout, this sort of notation will be used to generate formulas by substitution.) Of course each model of $state(F')$ also corresponds to a state in the obvious way. Finally, the formula

$$state(F) \wedge act(F, A, F') \wedge state(F') \tag{1}$$

encodes the set of transitions that correspond to its models: that is, each model of (1) encodes the transition such that (*i*) the start state corresponds to the interpretation of the symbols in F, (*ii*) the set of actions (concurrently) executed corresponds to the interpretation of the symbols in A, and (*iii*) the end state corresponds to the interpretation of the symbols in F'. We abbreviate (1) as

$$tr(F, A, F') \, .$$

This domain representation framework accomodates nondeterminism, concurrent actions and dynamic worlds. We do not address in this paper how such action representations are to be obtained. There are many possibilities. For instance, domain descriptions in STRIPS-based languages are easily translated (in polynomial time) into domain descriptions in the format used in this paper. Such descriptions can also be obtained from more expressive action languages, such as the high-level action language $\mathcal{C}$ [7], for which "definite" descriptions yield such classical propositional theories in polynomial time by the syntactic process of "literal completion" introduced in [12].

2.2 Planning Framework

In the remainder of the paper, we use this action representation framework to investigate the computational complexity of several classes of planning problems. That is, we consider the complexity of the associated decision problem—existence of a valid plan. The input size parameter for these complexity results is (any reasonable measure of) the size of (1) plus the size of the formulas $init(F)$ and $goal(F)$ describing, respectively, the initial state and the goal. (As before, we assume that the nonlogical symbols in these formulas are taken from F.) Henceforth, we will find it convenient to assume that $init(F) \models state(F)$. We will focus on plans whose length is bounded by a polynomial in the input size.

2.3 The Polynomial Hierarchy and PSPACE

For our purposes, it is convenient to characterize complexity classes in terms of second-order propositional logic. (See, for instance, [13].) Consider quantified boolean formulas (QBFs) of the form

$$Q_1 \, x_1 \; Q_2 \, x_2 \;\; \cdots \;\; Q_k \, x_k \;\, \Phi(x_1, x_2, \ldots, x_k) \tag{2}$$

where (i) each of $x_1, x_2, \ldots, x_k$ stands for a tuple of propositional variables, (ii) each of $Q_1, Q_2, \ldots, Q_k$ stands for a quantifier, with each Q_{i+1} different from Q_i, and (iii) $\Phi(x_1, x_2, \ldots, x_k)$ is a formula without quantifiers and without nonlogical constants, whose variables are taken from $x_1, x_2, \ldots, x_k$. For a formula of this kind, the corresponding decision problem is this: Is it satisfiable? Equivalently, we can ask if it is logically valid, or, since no nonlogical constants occur in it, we may simply ask if it is true. For natural number k, complexity class Σ_k^P corresponds to the family of formulas of form (2) with $Q_1 = \exists$, and complexity class Π_k^P corresponds to the family of such formulas with $Q_1 = \forall$. That is, for each of these families of QBFs, the decision problem is complete for the corresponding complexity class. Some important special cases are $\Sigma_1^P = \mathbf{NP}$, $\Pi_1^P = \mathbf{coNP}$, and $\Sigma_0^P = \Pi_0^P = \mathbf{P}$. Finally, the complexity class **PSPACE** corresponds to the decision problem for the family of all QBFs of form (2).

2.4 Some Common Assumptions in Planning

The following QBFs describe some common assumptions in planning.

$$\exists f \;\; (\mathit{init}(f) \wedge \forall f' \; (\mathit{init}(f') \supset f = f')) \tag{3}$$

$$\forall f \; \forall a \; \forall f' \; \forall f'' \; (\mathit{tr}(f, a, f') \wedge \mathit{tr}(f, a, f'') \supset f' = f'') \tag{4}$$

$$\forall f \; \forall a \; \exists f' \; (\mathit{state}(f) \supset \mathit{tr}(f, a, f')) \tag{5}$$

The assumption that the initial state is completely described (i.e. unique) is expressed by (3) where f (respectively f') is a tuple of distinct propositional variables corresponding to the propositional constants in F (respectively F'), and $f = f'$ is shorthand for the conjunction of formulas $p \equiv p'$ for each pair of similar variables $p \in f$ and $p' \in f'$. (From now on we'll employ such notation without remarking on it.) An action domain is deterministic iff (4) is true. So determining that an action domain is deterministic is a problem in **coNP**. Moreover, it is **coNP**-hard. Actions are always executable iff (5) is true. So determining that actions are always executable belongs to Π_2^P. Again, hardness is easily shown.

In many accounts of planning, it is unusual for actions to be always executable. Nonetheless, action description languages used for planning often have a similarly useful property: there are polynomial-time algorithms not only for obtaining formulas $\mathit{state}(F)$ and $\mathit{act}(F, A, F')$, but also for obtaining a classical propositional formula $\mathit{executable}(F, A)$ (in which only atoms from $F \cup A$ occur) that is logically equivalent to $\mathit{state}(F) \supset \exists f' \, \mathit{tr}(F, A, f')$. Such action description languages make it easy to determine whether actions are executable in a state. In fact, we note (without going into details) that in such cases there is a p-time transformation that yields a description for which actions are always executable and yet the valid plans (for all classes of planning problems considered here) are exactly the same. (Moreover, this can be done without introducing additional concurrency or nondeterminism.) Thus, for instance, a planning problem posed using a STRIPS-based action representation language can be translated in polynomial time into our planning framework, with the additional guarantee that actions will be always executable in the resulting domain description.

We say that the executability of actions is polynomial-time computable iff, for every interpretation I of $F \cup A$ that satisfies $\mathit{state}(F)$, deciding whether $I \models \exists f' \, \mathit{tr}(F, A, f')$ can be done in polynomial time. In our framework it is in general not the case that the executability of actions in a state can be determined in polynomial time (assuming $\mathbf{P} \neq \mathbf{NP}$); one easily verifies that the problem of determining that given actions are executable in a given state is **NP**-complete.

3 Classical Planning

For us, classical planning is essentially what is widely known as satisfiability planning [9]. In satisfiability planning, an action domain is (also) described by a finite theory of classical propositional logic. Typically, the language will include action atoms (representing the proposition that a certain action occurs at a certain time) and fluent atoms (representing the proposition that a certain fluent holds at a certain time). Assume that action atoms have the form α_t, where α is an action symbol and subscript t is a natural number—the "time stamp" of α_t. Similarly, assume that fluent atoms have the form φ_t, with φ a fluent symbol and t a time stamp. A plan of length k is obtained by finding a model of

$$\mathit{init}(F_0) \wedge \bigwedge_{t=0}^{k-1} \mathit{tr}(F_t, A_t, F_{t+1}) \wedge \mathit{goal}(F_k)$$

where each F_i (respectively A_i) is the set of fluent atoms (respectively action atoms) obtained by adding time stamp i to each fluent symbol (respectively action symbol). The plan is the sequence of (possibly concurrent) action occurrences corresponding to the interpretation of the action atoms. In order to guarantee that this plan is valid—that is, always executable and sure to achieve the goal under all possible executions—the initial state must be completely described, and the domain must be deterministic. If these assumptions are indeed satisfied, the existence of a valid plan is expressed by the QBF

$$\exists f_0 \; \overbrace{\exists a_0 \; \exists f_1} \; \cdots \; \overbrace{\exists a_{k-1} \; \exists f_k} \; \Big(\mathit{init}(f_0) \wedge \bigwedge_{t=0}^{k-1} \mathit{tr}(f_t, a_t, f_{t+1}) \wedge \mathit{goal}(f_k)\Big) \qquad (6)$$

where each f_i (respectively a_i) is a tuple of distinct propositional variables corresponding to the propositional constants in F_i (respectively A_i). This shows once again what is widely known: classical planning, for plans of polynomially-bounded length, belongs to **NP**.

Theorem 1. *Classical planning for plans of length 1 is* **NP***-hard, even if actions are always executable. Classical planning is* **NP***-complete for plans of polynomially-bounded length.*

Recall though that, in the current framework, classical planning includes the possibility of concurrent actions and dynamic worlds. Moreover, it may not be possible to determine in polynomial time whether given actions are executable in a given state. These complications do not push the problem beyond **NP**.

4 Conformant Planning

In conformant planning, a plan is still a sequence of actions, but the action domain may be nondeterministic, and the initial state may be incompletely specified. As usual, a plan is valid if it is guaranteed to be executable, and every possible execution achieves the goal. The key issues can be illustrated by considering plans of length 1. There is at least one possible initial state:

$$\exists f_0 \ \mathit{init}(f_0)\,. \tag{7}$$

There is an unconditional one-step plan that is executable starting from any of the possible initial states:

$$\exists a_0 \ \forall f_0 \ (\mathit{init}(f_0) \supset \exists f_1 \ \mathit{tr}(f_0, a_0, f_1))\,. \tag{8}$$

There is an unconditional one-step plan that achieves the goal in every possible execution starting from any of the possible initial states:

$$\exists a_0 \ \forall f_0 \ (\mathit{init}(f_0) \supset \forall f_1 \ (\mathit{tr}(f_0, a_0, f_1) \supset \mathit{goal}(f_1)))\,.$$

So to say that there is a valid one-step unconditional plan—one that is both executable (starting in any possible initial state) and sufficient to achieve the goal (when executed starting in any possible initial state)—we can write

$$\exists a_0 \ \forall f_0 \ (\mathit{init}(f_0) \supset \exists f_1 \ \mathit{tr}(f_0, a_0, f_1) \land \ \forall f_1 \ (\mathit{tr}(f_0, a_0, f_1) \supset \mathit{goal}(f_1)))\,. \tag{9}$$

Consequently, the decision problem for one-step unconditional planning is expressed by the conjunction of (7) and (9). We can put (9) in prenex form as follows: $\exists a_0 \ \forall f_0 \ \forall f_1 \ \exists f_1' \ (\mathit{init}(f_0) \supset \mathit{tr}(f_0, a_0, f_1') \land (\mathit{tr}(f_0, a_0, f_1) \supset \mathit{goal}(f_1)))\,.$ Hence, the decision problem for one-step conformant planning belongs to the complexity class Σ_3^P. Hardness is easily shown.

As a step toward the general formulation, consider plans of length 2. We need not alter the formula (7) expressing the existence of at least one possible initial state. To say that there is an unconditional plan whose first step is executable in any possible initial state, we can (again) write (8). To say that there is an unconditional plan such that the second step is executable in any state reached after executing the first step in any possible initial state, we can write

$$\exists a_0 \ \exists a_1 \ \forall f_0 \ \forall f_1 (\mathit{init}(f_0) \land \mathit{tr}(f_0, a_0, f_1) \supset \exists f_2 \ \mathit{tr}(f_1, a_1, f_2))\,. \tag{10}$$

To say that there is an unconditional two-step plan whose execution is sufficient to achieve the goal (starting in any possible initial state), we can write

$$\exists a_0 \ \exists a_1 \ \forall f_0 \ \forall f_1 \ \forall f_2 \ (\mathit{init}(f_0) \land \mathit{tr}(f_0, a_0, f_1) \land \mathit{tr}(f_1, a_1, f_2) \supset \mathit{goal}(f_2))\,. \tag{11}$$

Combining (8), (10) and (11) appropriately in prenex form, we can obtain

$$\begin{aligned}\exists a_0 \ \exists a_1 \ \forall f_0 \ \forall f_1 \ \forall f_2 \ \exists f_1' \ \exists f_2' \ ((&\mathit{init}(f_0) \supset \mathit{tr}(f_0, a_0, f_1'))\\ \land\ (&\mathit{init}(f_0) \land \mathit{tr}(f_0, a_0, f_1) \supset \mathit{tr}(f_1, a_1, f_2'))\\ \land\ (&\mathit{init}(f_0) \land \mathit{tr}(f_0, a_0, f_1) \land \mathit{tr}(f_1, a_1, f_2) \supset \mathit{goal}(f_2)))\,.\end{aligned}$$

The general form is similar. For conformant planning for plans of length k, take the conjunction of (7) and

$$\begin{aligned}\exists a_0 \cdots \exists a_{k-1}\ \forall f_0 \cdots \forall f_k\ \exists f'_1 \cdots \exists f'_k \\ \Big(\textstyle\bigwedge_{t=0}^{k-1} \Big(\mathit{init}(f_0) \wedge \textstyle\bigwedge_{u=0}^{t-1} \mathit{tr}(f_u, a_u, f_{u+1}) \supset \mathit{tr}(f_t, a_t, f'_{t+1}) \Big) \\ \wedge \Big(\mathit{init}(f_0) \wedge \textstyle\bigwedge_{t=0}^{k-1} \mathit{tr}(f_t, a_t, f_{t+1}) \supset \mathit{goal}(f_k) \Big) \Big) .\end{aligned} \tag{12}$$

Theorem 2. *Conformant planning for plans of length 1 is Σ_3^P-hard. Conformant planning is Σ_3^P-complete for plans of polynomially-bounded length.*

In a comment following their formally stated complexity results, Eiter *et al.* [5] mention a similar Σ_3^P-completeness result for conformant planning, but their result is restricted to plans of a fixed length. They employ an expressive high-level action description language whose salient feature in this context is that determining the executability of given actions in a given state is **NP**-complete.

If we assume that actions are always executable, we can eliminate the check for plan executability from (12), which can then be reduced to

$$\exists a_0 \cdots \exists a_{k-1}\ \forall f_0 \cdots \forall f_k\ \Big(\mathit{init}(f_0) \wedge \textstyle\bigwedge_{t=0}^{k-1} \mathit{tr}(f_t, a_t, f_{t+1}) \supset \mathit{goal}(f_k) \Big) .$$

So the complexity of conformant planning falls to Σ_2^P if actions are always executable. We can then show hardness for plans of length 1 even if we assume determinism. On the other hand, we can show membership even if the executability of given actions in a given state is only polynomial-time decidable.

Theorem 3. *Conformant planning for plans of length 1 is Σ_2^P-hard for deterministic planning domains in which actions are always executable. Conformant planning is Σ_2^P-complete for plans of polynomially-bounded length, if the executability of actions is polynomial-time computable.*

A similar result (Theorem 2) in [2] is obtained using an action representation language that guarantees determinism and nonconcurrency, and makes it easy to determine whether actions are executable in a state (as discussed in Section 2.4). Our analysis suggests that only the last property is crucial for placing the problem within Σ_2^P. On the other hand, as mentioned previously, [5] employs a relatively expressive action description language, and a related Σ_2^P result reported there (Theorem 3) is also obtained by directly imposing polynomial-time decidability of executability of given actions in a given state. The latter result again assumes that plan length is fixed (not just polynomially-bounded).

5 Conditional Planning

Conditional planning is an extension of conformant planning in which the actions to be performed in executing a plan may depend on the previous execution history. For the analysis in this section, we assume that each step in the execution of a conditional plan may depend arbitrarily on the previous execution history. Thus, in a rather general sense, we address here the fundamental question of existence of a valid conditional plan.

As with conformant planning, it is convenient to begin by considering plans of length 1. Of course the existence of at least one possible initial state is again expressed by (7). To say that there is a one-step conditional plan that is executable in any possible initial state, we can write

$$\forall f_0 \ (\mathit{init}(f_0) \supset \exists a_0 \ \exists f_1 \ \mathit{tr}(f_0, a_0, f_1))\,. \tag{13}$$

To say that there is a one-step conditional plan that is sufficient to achieve the goal if executed starting in any possible initial state, we can write

$$\forall f_0 \ (\mathit{init}(f_0) \supset \exists a_0 \ \forall f_1 \ (\mathit{tr}(f_0, a_0, f_1) \supset \mathit{goal}(f_1)))\,.$$

Consequently, a formula for the decision problem corresponding to one-step conditional planning is the conjunction of (7) and

$$\forall f_0 \ \exists a_0 \ \exists f_1' \ \forall f_1 (\mathit{init}(f_0) \supset \mathit{tr}(f_0, a_0, f_1') \wedge (\mathit{tr}(f_0, a_0, f_1) \supset \mathit{goal}(f_1)))\,.$$

As a step toward obtaining the general form, let's consider conditional plans of length 2. We need not alter the formula (7) expressing the existence of at least one possible initial state. To say that there is a conditional plan whose first step is executable in any possible initial state, we (again) write (13). To say that there is a conditional plan such that the second step is executable in any state reached after executing the first step in any possible initial state, we write

$$\forall f_0 \ (\mathit{init}(f_0) \supset \exists a_0 \ \forall f_1 \ (\mathit{tr}(f_0, a_0, f_1) \supset \exists a_1 \ \exists f_2 \ \mathit{tr}(f_1, a_1, f_2)))\,.$$

To say that there is a two-step conditional plan whose execution is sufficient to achieve the goal (starting in any possible initial state), we can write

$$\forall f_0 \ (\mathit{init}(f_0) \supset \exists a_0 \ \forall f_1 \ (\mathit{tr}(f_0, a_0, f_1) \supset \exists a_1 \ \forall f_2 \ (\mathit{tr}(f_1, a_1, f_2) \supset \mathit{goal}(f_2))))\,.$$

Combining (13) and these last two formulas in prenex form, we can obtain

$$\begin{aligned} \forall f_0 \ \exists a_0 \ \forall f_1 \ \exists a_1 \ \exists f_1' \ \exists f_2' \ \forall f_2 \ & ((\mathit{init}(f_0) \supset \mathit{tr}(f_0, a_0, f_1')) \\ & \wedge (\mathit{init}(f_0) \wedge \mathit{tr}(f_0, a_0, f_1) \supset \mathit{tr}(f_1, a_1, f_2')) \\ & \wedge (\mathit{init}(f_0) \wedge \mathit{tr}(f_0, a_0, f_1) \wedge \mathit{tr}(f_1, a_1, f_2) \supset \mathit{goal}(f_2)))\,. \end{aligned}$$

Written in this form, it is clear that the "choice" of actions to be performed at time 0 depends only on the values of fluents at time 0. Similarly, the choice of actions to be performed at time 1 depends only on the values of fluents at time 0, the actions performed at time 0, and the values of fluents at time 1.

Here's the general form. For conditional planning for plans of length k, take the conjunction of (7) and the following.

$$\begin{aligned} & \overbrace{\forall f_0 \, \exists a_0} \cdots \overbrace{\forall f_{k-1} \, \exists a_{k-1}} \ \exists f_1' \cdots \exists f_k' \ \forall f_k \\ & \qquad \Big(\textstyle\bigwedge_{t=0}^{k-1} \Big(\mathit{init}(f_0) \wedge \textstyle\bigwedge_{u=0}^{t-1} \mathit{tr}(f_u, a_u, f_{u+1}) \supset \mathit{tr}(f_t, a_t, f_{t+1}') \Big) \\ & \qquad \wedge \Big(\mathit{init}(f_0) \wedge \textstyle\bigwedge_{t=0}^{k-1} \mathit{tr}(f_t, a_t, f_{t+1}) \supset \mathit{goal}(f_k) \Big) \Big) \end{aligned} \tag{14}$$

If actions are always executable, (14) can be simplified:

$$\forall f_0 \; \overbrace{\exists a_0 \forall f_1} \cdots \overbrace{\exists a_{k-1} \forall f_k} \; \Big(init(f_0) \wedge \textstyle\bigwedge_{t=0}^{k-1} tr(f_t, a_t, f_{t+1}) \supset goal(f_k) \Big) \, .$$

Theorem 4. *Conditional planning for plans of length k is Π^P_{2k+1}-hard, even if actions are always executable. Conditional planning for plans of length k ($k > 0$) is Π^P_{2k+1}-complete. Conditional planning is* **PSPACE**-*complete for plans of polynomially-bounded length.*

A related **PSPACE**-completeness result is due to Littman [10] (Theorem 2). Although concerned with probabilistic planning, his proof construction for hardness uses just nondeterminism. Littman's result does not imply the **PSPACE** membership result in Theorem 4; in his approach actions are always executable.

If the action domain is deterministic, (14) can be equivalently replaced with

$$\forall f_0 \; \overbrace{\exists a_0 \; \exists f_1} \cdots \overbrace{\exists a_{k-1} \; \exists f_k} \; \Big(init(f_0) \supset \textstyle\bigwedge_{t=0}^{k-1} tr(f_t, a_t, f_{t+1}) \wedge goal(f_k) \Big)$$

which reduces the complexity to Π^P_2! One can show hardness in this case even if actions are assumed to be always executable.

Theorem 5. *Conditional planning for plans of length 1 is Π^P_2-hard for deterministic planning domains, even if actions are always executable. Conditional planning is Π^P_2-complete for deterministic planning domains for plans of polynomially-bounded length.*

A related hardness result is obtained by Rintanen [14] (Theorem 2), using a deterministic action language, without concurrency, for which it is easy to determine executability of actions in a state.[1] A similar completeness result appears in [2] (Theorem 8), restricted, as mentioned previously, to deterministic action domains for which it is easy to determine executability of actions in a state. Our analysis suggests that the key is determinism.

6 Conditional Planning with Partial Observability

Despite the fact that Baral *et al.* in [2] consider only deterministic action domains, their Theorem 4 identifies a variant of polynomial-length conditional planning that is **PSPACE**-hard. This is because their action description language incorporates a rudimentary account of sensing actions, which allows them to model planning domains in which knowledge of the current state during plan execution is incomplete because the current state is only partially observable.

[1] The definition of planning used in Rintanen's theorem appears to subsume classical propositional STRIPS planning, and the theorem does not restrict consideration to bounded-length plans. Hence, it seems that **PSPACE**-hardness could have been derived immediately based on Bylander's well-known result [3]. On the other hand, Rintanen's proof construction utilizes only plans of polynomially-bounded length.

Such planning domains can be modeled in our setting by partitioning the set F of fluent symbols into two sets, F^O and F^N, intuitively corresponding to observable and nonobservable fluents, respectively.

Let's start again with plans of length 1. As usual, the existence of at least one possible initial state is expressed by (7). The existence of a conditional one-step plan that can be executed in any possible initial state is expressed by

$$\forall f_0^O\ \exists a_0\ \forall f_0^N\ (\mathit{init}(f_0) \supset \exists f_1\ \mathit{tr}(f_0, a_0, f_1))\,.$$

Notice that the choice of actions to be performed at time 0 depends only on the values of *observable* fluents at time 0. The existence of a conditional one-step plan sufficient to achieve the goal (when executed starting in any possible initial state) is expressed by

$$\forall f_0^O\ \exists a_0\ \forall f_0^N\ (\mathit{init}(f_0) \supset \forall f_1\ (\mathit{tr}(f_0, a_0, f_1) \supset \mathit{goal}(f_1)))\,.$$

Again, the choice of actions to be performed at time 0 depends only on the values of observable fluents at time 0. As we did previously for simpler forms of planning, we combine formulas expressing executability and sufficiency, obtaining

$$\forall f_0^O\ \exists a_0\ \forall f_0^N\ \forall f_1\ \exists f_1'\ (\mathit{init}(f_0) \supset \mathit{tr}(f_0, a_0, f_1') \wedge (\mathit{tr}(f_0, a_0, f_1) \supset \mathit{goal}(f_1)))\,,$$

which, conjoined with (7), expresses the decision problem for one-step conditional planning with partial observability.

To extend this to two-step conditional planning, we need to express the existence of a two-step conditional plan such that the second step is executable in any state reached by executing the first step in any possible initial state. In doing so, we must capture the fact that (i) the choice of actions to be performed at time 0 depends only on the observable fluents at time 0, and (ii) the choice of actions to be performed at time 1 depends only on the observable fluents at time 0, the actions performed at time 0, and the observable fluents at time 1.

$$\forall f_0^O\ \exists a_0\ \forall f_1^O\ \exists a_1\ \forall f_0^N\ \forall f_1^N\ (\mathit{init}(f_0) \wedge \mathit{tr}(f_0, a_0, f_1) \supset \exists f_2\ \mathit{tr}(f_1, a_2, f_2))$$

We similarly need to express that there exists such a plan that is sufficient to achieve the goal when executed starting in any possible initial state.

$$\forall f_0^O\ \exists a_0\ \forall f_1^O\ \exists a_1\ \forall f_0^N\ \forall f_1^N\ (\mathit{init}(f_0) \wedge \mathit{tr}(f_0, a_0, f_1) \supset \forall f_2\ (\mathit{tr}(f_1, a_2, f_2) \supset \mathit{goal}(f_2)))$$

These observations suggest that two-step conditional planning with partial observability can be expressed by the conjunction of (7) and

$$\begin{aligned}\forall f_0^O\ \exists a_0\ \forall f_1^O\ \exists a_1\ \forall f_0^N\ \forall f_1^N\ \forall f_2\ \exists f_1'\ \exists f_2'\ &((\mathit{init}(f_0) \supset \mathit{tr}(f_0, a_0, f_1'))\\ &\wedge\ (\mathit{init}(f_0) \wedge \mathit{tr}(f_0, a_0, f_1) \supset \mathit{tr}(f_1, a_1, f_2'))\\ &\wedge\ (\mathit{init}(f_0) \wedge \mathit{tr}(f_0, a_0, f_1) \wedge \mathit{tr}(f_1, a_1, f_2) \supset \mathit{goal}(f_2)))\,.\end{aligned}$$

Here's the general form. For conditional planning with partial observability for plans of length k, take the conjunction of (7) and the following.

$$\begin{aligned}\overbrace{\forall f_0^O\, \exists a_0}\ \cdots\ \overbrace{\forall f_{k-1}^O\, \exists a_{k-1}}\ \forall f_0^N \cdots \forall f_{k-1}^N\, \forall f_k\, \exists f_1' \cdots \exists f_k'\qquad&\\ \Big(\textstyle\bigwedge_{t=0}^{k-1} \Big(\mathit{init}(f_0) \wedge \bigwedge_{u=0}^{t-1} \mathit{tr}(f_u, a_u, f_{u+1}) \supset \mathit{tr}(f_t, a_t, f_{t+1}')\Big)&\\ \wedge\ \Big(\mathit{init}(f_0) \wedge \textstyle\bigwedge_{t=0}^{k-1} \mathit{tr}(f_t, a_t, f_{t+1}) \supset \mathit{goal}(f_k)\Big)\Big)&\end{aligned} \tag{15}$$

Theorem 6. *Conditional planning with partial observability for plans of length k $(k > 0)$ is Π^P_{2k+2}-complete. Conditional planning with partial observability is* **PSPACE**-*complete for plans of polynomially-bounded length.*

If actions are always executable, take the conjunction of (7) and the following.

$$\overbrace{\forall f^O_0 \, \exists a_0} \cdots \overbrace{\forall f^O_{k-1} \, \exists a_{k-1}} \, \forall f^N_0 \cdots \forall f^N_{k-1} \, \forall f_k \, \Big(init(f_0) \wedge \textstyle\bigwedge_{t=0}^{k-1} tr(f_t, a_t, f_{t+1}) \supset goal(f_k) \Big)$$

So the problem falls to Π^P_{2k+1}. We can show hardness even with determinism.

Theorem 7. *For deterministic planning domains with actions always executable, conditional planning with partial observability is Π^P_{2k+1}-hard for plans of length k, and so* **PSPACE**-*hard for plans of polynomially-bounded length. Conditional planning with partial observability for plans of length k $(k > 0)$ is Π^P_{2k+1}-complete, if the executability of actions is polynomial-time computable.*

7 Effect of No Concurrent Actions, Fixed Plan Length

Eliminating concurrency has little effect on hardness. The previous theorems establish hardness results for plans of fixed length, but the proof constructions rely on concurrency. Corresponding hardness results for nonconcurrent domains can be established by allowing instead plans of length $|A|$ (recall that A is the set of action symbols), so plan length can be linearly-bounded in the size of input. (In each case, there is a corresponding proof construction in which actions previously performed concurrently are performed sequentially instead.)

We have noted that the complexity results from [5] most closely related to those of this paper employ an assumption of fixed plan length (rather than polynomially-bounded plan length). It may be interesting to consider the effect of assuming both nonconcurrency and fixed plan length.

For classical and conformant planning, a fixed plan length guarantees that the number of possible plans with no concurrent actions is polynomially-bounded: there are $(|A| + 1)^k$ nonconcurrent plans of length k. Consequently, the existential quantifiers over actions in formula (6) for classical planning and formula (12) for conformant planning can be eliminated in polynomial time.

It is not clear that this possibility reduces the complexity of classical planning. Roughly speaking, after quantifiers over actions are eliminated from (6), we are left with a disjunction of a polynomially many "unambiguous sat" problems (which ask: Is a propositional theory with at most one model satisfiable?). This problem is thought to be beyond **P** without being **NP**-hard [13].

On the other hand, when plan length is fixed the complexity of conformant planning drops for nonconcurrent domains.

Theorem 8. *For nonconcurrent domains:*

(i) Conformant planning for plans of length 1 is Π^P_2-hard.
(ii) Conformant planning for plans of fixed length is Π^P_2-complete.

(*iii*) *For deterministic planning domains in which actions are always executable, conformant planning for plans of length 1 is D^P-hard.*[2]

(*iv*) *For deterministic planning domains in which the executability of actions is polynomial-time computable, conformant planning for plans of fixed length is D^P-complete.*

Parts (*ii*) and (*iv*) of Theorem 8 are very similar to results in [5] (Theorem 4 and subsequent remarks).

What about the effect of fixed plan length on the complexity of conditional planning without concurrency? It is clear that we do not obtain a polynomial bound on the number of possible plans. In fact, we do not even obtain a polynomial bound on the space needed to represent a conditional plan, since a conditional plan is essentially a function that maps the observable portion of a partial plan execution to the next action to be performed.

8 Conditional Planning with Polynomially-Bounded Plan Representations

Since even a fixed plan length and an assumption of nonconcurrency does not yield a polynomial bound on the size of conditional plans, we may wish to consider imposing such a bound directly. In fact this possibility is crucial to the computational approach to conditional planning described by Rintanen [14]. As mentioned previously, Rintanen's paper includes a proof of Π_2^P-hardness for conditional planning. But his main concern is with computational approaches based on problem encodings that directly restrict the size of possible plans to be considered. More precisely, he introduces several different planning problem encodings (in second-order propositional logic), each requiring only polynomial time to construct (given a domain description in his action representation language). Rintanen claims for such computational approaches a complexity of Σ_2^P. Notice though that we can understand conformant planning as perhaps the simplest possible manifestation of such an approach, and *it* is Σ_3^P-hard. It appears that Rintanen's informal argument depends on an implicit assumption that executability of actions in a state is polynomial-time computable.

Rintanen's approach resembles that of Littman *et al.* [11] in the setting of probabilistic planning (with the implicit assumption that actions are always executable). They consider several plan encodings of polynomially-bounded size and obtain complexity results intuitively related to the Σ_2^P result for propositional conditional planning (but instead of $\Sigma_2^P = \mathbf{NP}^{\mathbf{NP}}$ the complexity is $\mathbf{NP}^{\mathbf{PP}}$, which is still within **PSPACE**, but contains all of the polynomial hierarchy).

It appears that such results are necessarily fragmentary—that is, each depends on a particular choice of (polynomial-size) plan representation—because, unlike the case of k-length conditional plans, for instance, the set of k-size conditional plans apparently cannot be defined independent of a choice of plan

[2] The complexity class D^P consists of problems that can be solved in polynomial-time given the answer to one **NP** problem and one **coNP** problem.

representation. Presumably, not just any polynomial-size plan representation will do. In particular, one expects a plan representation that guarantees that the plan function—mapping partial execution histories to next actions—is computable. In fact, it seems reasonable to choose a representation guaranteed to allow polynomial-time computation of the plan function (as it seems Littman *et al.* and Rintanen do in all cases).

As with conformant planning, it appears that, for suitable choices of plan representation, the problem of existence of a valid conditional plan of polynomially-bounded size will fall in complexity class Σ_3^P, and will drop from Σ_3^P to Σ_2^P if the executability of actions is polynomial-time computable.

Acknowledgements

Thanks to Chitta Baral, Vladimir Lifschitz, Jussi Rintanen and a few of the (rather many) anonymous referees for helpful comments on various drafts. This work partially supported by NSF Career Grant #0091773.

References

1. Christer Bäckström and Bernhard Nebel. Complexity results for SAS+ planning. *Computational Intelligence*, 11(4):625–655, 1995.
2. Chitta Baral, Vladik Kreinovich, and Raul Trejo. Computational complexity of planning and approximate planning in presence of incompleteness. *Artificial Intelligence*, 122(1–2):241–267, 2000.
3. Tom Bylander. The computational complexity of propostional STRIPS planning. *Artificial Intelligence*, 69(1–2):165–204, 1994.
4. Giuseppe De Giacomo and Moshe Vardi. Automata-theoretic approach to planning for temporally extended goals. In *Proc. of 5th European Conf. on Planning*, 1999.
5. Thomas Eiter, Wolfgang Faber, Nicola Leone, Gerald Pfeifer, and Axel Polleres. Planning under incomplete information. In *Proc. Computational Logic 2000*, 2000.
6. Kutluhan Erol, Dana S. Nau, and V.S. Subrahmanian. Complexity, decidability and undecidability results for domain-independent planning. *Artificial Intelligence*, 76(1–1):75–88, 1995.
7. Enrico Giunchiglia and Vladimir Lifschitz. An action language based on causal explanation: Preliminary report. In *Proc. AAAI-98*, pages 623–630, 1998.
8. Patrik Haslum and Peter Jonsson. Some results on the complexity of planning with incomplete information. In *Proc. of 5th European Conf. on Planning*, 1999.
9. Henry Kautz and Bart Selman. Planning as satisfiability. In *Proc. of the 10th European Conf. on Artificial Intelligence*, pages 359–379, 1992.
10. Michael Littman. Probabilistic propositional planning: representations and complexity. In *Proc. of AAAI-97*, pages 748–754, 1997.
11. Michael Littman, Judy Goldsmith, and Martin Mundhenk. The computational complexity of probabilistic planning. *Journal of Artificial Intelligence Research*, 9:1–36, 1998.
12. Norman McCain and Hudson Turner. Causal theories of action and change. In *Proc. of AAAI-97*, pages 460–465, 1997.
13. Christos Papadimitriou. *Computational Complexity.* Addison Wesley, 1994.
14. Jussi Rintanen. Constructing conditional plans by a theorem prover. *Journal of Artificial Intelligence Research*, 10:323–352, 1999.