Problem Solving Using Classical Planners
AAAI Technical Report WS-12-12

Making Reasonable Assumptions to Plan with Incomplete Information: Abridged Report*

Sammy Davis-Mendelow
Department of Computer Science
University of Toronto
Toronto, Canada

Jorge A. Baier
Depto. de Ciencia de la Computación
Pontificia Universidad Católica de Chile
Santiago, Chile

Sheila A. McIlraith
Department of Computer Science
University of Toronto
Toronto, Canada

Abstract

Many practical planning problems necessitate the generation of a plan under incomplete information about the state of the world. In this paper we propose the notion of Assumption-Based Planning. Unlike conformant planning, which attempts to find a plan under all possible completions of the initial state, an assumption-based plan supports the assertion of additional assumptions about the state of the world, simplifying the planning problem. In many practical settings, such plans can be of higher quality than conformant plans. We formalize the notion of assumption-based planning, establishing a relationship between assumption-based and conformant planning, and prove properties of such plans. We further provide for the scenario where some assumptions are more preferred than others. Exploiting the correspondence with conformant planning, we propose a means of computing assumption-based plans via a translation to classical planning. Our translation is an extension of the popular approach proposed by Palacios and Geffner and realized in their T0 planner. We have implemented our planner, A0, as a variant of T0 and tested it on a number of expository domains drawn from the International Planning Competition. Our results illustrate the utility of this new planning paradigm.

Introduction

In many real-world planning problems, only a subset of the state of the world may be known. Conformant planning, conditional planning, probabilistic planning and contingent planning are among the approaches used to address such planning scenarios. Whereas classical planning assumes complete information about the state of the world, conformant planning assumes incomplete information but necessitates generation of a plan that relies only on what is known. This makes planning difficult and can lead to poor plans.

In this paper we define the notion of *assumption-based planning*. Assumption-based planning attempts to find a middle-ground between conformant and classical planning wherein the planner dynamically asserts reasonable, calculated assumptions about the uncertainty in the world in order to generate a valid plan given those assumptions. In contrast to contingent planning that exploits strategic sensing to resolve uncertainty, assumption-based planning is well-suited to scenarios where resolving uncertainty directly is impossible, difficult, or expensive.

The term assumption-based planning has been coined for a number of diverse planning activities that broadly relate to assumptions. Albore and Bertoli (2004; 2006) used the term to describe a notion of planning in which an assumption is provided a priori as a linear temporal logic (LTL) formula and a plan is generated predicated on this assumption. Their work is more closely related to planning with LTL domain control knowledge (e.g., Bacchus and Kabanza 2000). Pellier and Fiorino (2004; 2005) also use the term to describe a multi-agent approach to devising a shared global plan via a conjecture/refutation cycle, where agents exchange proposals and counter-proposals in an argumentation dialogue.

Here, assumption-based planning is related to the characterization of abduction as theory formation (e.g., Poole, Goebel, and Aleliunas 1987) wherein additional facts about the world are conjectured in order to explain an observation. It is also somewhat related to the notion of generating explanations for dynamical systems (e.g., Sohrabi, Baier, and McIlraith 2011). Indeed, Reiter and de Kleer (1987) established the relationship between abduction (explanation generation) and assumption-based reasoning for static systems, and Conrad and Williams (2011) employed aspects of assumption-based truth maintenance in their Drake executive for temporal plans. In (Göbelbecker, Gretton, and Dearden 2011; Bonet and Geffner 2011; Albore and Geffner 2009), contingent planners may make assumptions that can be verified through sensing later on.

In contrast to previous work, we provide a formal characterization of assumption-based planning establishing a correspondence to conformant planning. We show how assumption-based plans can be computed via classical planning, proving the correctness of our approach. We also argue for the merit of preferred assumption-based plans and propose a means of realizing such plans via cost-based planning. We implement these two approaches and illustrate and assess some of their properties.

Characterization

In this section we formally define assumption-based planning and initial state assumption-based planning, as well as state the equivalence of the two given deterministic actions and no exogenous events.

*An unabridged version of this paper appeared in the HSDIP workshop of the ICAPS-2012 conference.

Background

We now define the fundamental notions that will be used in the rest of the paper. Our first definition is for *planning problems*.

Definition 1 (Planning Problem) *A planning problem is a tuple $P = (F, O, I, G)$ where F is a finite set of fluent symbols, O is a finite set of action operators, I is a set of clauses over F, defining the set of possible initial states, and G is a boolean formula over symbols in F, that defines a goal condition.*

Every action $a \in O$ is defined by a conjunction of fluent literals, $prec(a)$ (preconditions) and a set of conditional effects $C \rightarrow L$ where L is a fluent literal that is made true when the action is executed and the conjunction of fluent literals C holds.

Example 1 Consider a car-driving domain in which a car can drive between locations. There are three actions: $drive(x, y)$, $turnOn$, and $turnOff$. There are only two locations: $home$ and $office$. Initially we only know that the car is at home and that its engine is not on. Hence, the initial state I is given by $\{at(home), \neg at(office), \neg engineOn\}$. Action $turnOn$ turns the engine on if the battery is working, represented by the conditional effect $\{batteryOk \rightarrow engineOn\}$, and has no preconditions (i.e., $prec(a) = \mathsf{true}$). Action $drive(x, y)$ requires as precondition that $at(x)$ and that $engineOn$, and has the (unconditional) effects $at(y)$, $\neg at(x)$. Action $fixBattery$ has no precondition and a single conditional effect $\neg batteryOk \rightarrow batteryOk$. Finally, the objective is to bring the car to the office: $G = at(office)$.

A *planning state* s is defined by a set of fluent symbols, which represent all that is true. Each system state s induces a propositional valuation $M_s : F \rightarrow \{\mathsf{true}, \mathsf{false}\}$ that maps any fluent literal in s to true, and all other literals to false.

We say a state s is *consistent* with a set of clauses $\mathcal{C}$, if $M_s \models c$, for every $c \in \mathcal{C}$. Intuitively, $M_s \models \phi$ stands for "boolean formula ϕ holds true in state s".

An action a is *executable* in a state s if $M_s \models prec(a)$. If a is executable in a state s, we define its successor state as $\delta(a, s) = (s \setminus Del) \cup Add$, where Add contains a fluent f iff $C \rightarrow f$ is an effect of a and $M_s \models C$. On the other hand Del contains a fluent f iff $C \rightarrow \neg f$ is an effect of a, and $M_s \models C$. We define $\delta(a_0 a_1 \ldots a_n, s) = \delta(a_1 \ldots a_n, \delta(a_0, s))$, and $\delta(\epsilon, s) = s$. A sequence of actions α is *executable* in s if $\delta(\alpha, s)$ is defined. Furthermore α is executable in P iff it is executable in s, for any s consistent with I.

Below we define the notion of *execution trace* which intuitively characterizes maximal state trajectories that could result from the execution of an action sequence when performed in some of the possible initial states of a planning problem.

Definition 2 (Execution Trace) *A sequence of planning states $\sigma = s_0 s_1 \cdots s_k$ is an* execution trace *of $\alpha = a_0 a_1 \ldots a_n$ in planning problem $P = (F, O, I, G)$ iff (1) s_0 is consistent with I, (2) $\delta(a_i, s_i) = s_{i+1}$, for all $i < k$, and (3) either $k = n + 1$ or $k < n + 1$ and $\delta(s_k, a_k)$ is undefined.*

Definition 3 (Successful Execution Trace) *An execution trace σ for α is successful iff $|\sigma| = |\alpha| + 1$.*

Naturally, we are interested in execution traces that lead to goal satisfaction, i.e., for which the goal holds in the final state of the sequence of planning states. Formally,

Definition 4 (Leads to) *An execution trace $\sigma = s_0 \cdots s_k$ leads to (goal formula) G, iff $M_{s_k} \models G$.*

With the previous definitions in hand, we define the standard notion of conformant plan.

Definition 5 (Conformant Plan) *A sequence of actions α is a conformant plan for $P = (F, O, I, G)$ iff every execution trace of α is successful and leads to G.*

In our car-driving example, the sequence $fixBattery; turnOn; drive(home, office)$ is a conformant plan. The reader can easily verify that action $fixBattery$ is needed in any conformant plan.

Assumption-Based Planning

Consider an extension of our car-driving example where, in addition to the battery, many other car components are modeled as potentially malfunctioning. In the absence of information regarding the status of each component, a conformant plan would require fixing each component, which would result in either a long, poor quality plan, or possibly no plan at all. A contingent plan could be similarly poor, requiring significant computation or contingencies for unlikely scenarios. Instead we would like the planning system to simplify the task, as people do, by automatically assuming, in the absence of evidence to the contrary, that the battery and other components are functioning correctly in the initial state. Later on, we would like the planner to be capable of assuming that the freeway is not blocked in the state immediately before entering it. In general, we would like the planner to be able to make reasonable assumptions about *any* state along the execution.

Given a planning problem with an incomplete initial state, the task of assumption-based planning requires computing two elements: (1) a set of assumptions that are made at different states during the execution of the plan, and (2) a sequence of actions that, given the assumptions, is guaranteed to reach the goal. As such, the main difference between assumption-based planning and conformant planning is related to the computation of assumptions in addition to the computation of a plan.

Formally an *assumption-based planning problem* is a tuple $P = (F, O, I, G, T)$, where F, O, I, and G are defined exactly as for regular planning problems (Definition 1), and T is a subset of F and denotes the set of *assumable fluents*. T may be equivalent to the set of domain fluents or it may be restricted to an application-specific subset, such as the set of fluents corresponding to the normal functioning of car components in our car example. An assumption-based plan is a pair (ρ, α) where $\alpha = a_0 \cdots a_k$ is a sequence of actions and $\rho = h_0 \cdots h_{k+1}$ is a sequence of boolean formulae. Each h_i is a boolean formula over the fluents in T and represents assumptions made about the i-th state visited when performing α.

The execution traces that we will be interested in are those that *conform to* ρ; i.e., are such that they are consistent with the assumptions. Formally,

Definition 6 (Conforms to) *An execution trace* $\sigma = s_1 \cdots s_k$ *conforms to a sequence of boolean formulae* $\rho = h_1 \cdots h_n$ *with* $k \leq n$ *iff* $M_{s_i} \models h_i$*, for every* $i \in \{1, \ldots, k\}$.

Finally, each of the execution traces of α that conform to ρ must actually lead to the goal. A formal definition of an assumption-based plan follows.

Definition 7 (Assumption-Based Plan) *The pair* (ρ, α)*, where* α *is a sequence of* k *actions, and* ρ *is a sequence of* $k+1$ *boolean formulae over* T *is an assumption-based plan for* $P = (F, O, I, G, T)$ *iff any execution trace of* α *that conforms to* ρ *is successful and leads to* G*, and furthermore at least one such execution trace exists.*

Intuitively for every consistent completion of the initial state the execution trace is either successful and leads to the goal or is pruned by ρ. Assumption-based planning may be reduced to a conformant plan when all assumptions are restricted to be trivial. An assumption is trivial when it is entailed by the state in which it is made. Trivial assumptions may be forced by restricting the set T, the extreme case being when T is empty.

In our car example, $\pi = (\rho, \alpha)$, with $\rho = batteryOk; \mathtt{true}; \mathtt{true}$, $\alpha = turnOn; drive(home, office)$ is an assumption-based plan that assumes the battery is initially working and uses two actions to achieve the goal.

An important fact to note at this point is that, given the current definition of the problem, assumptions may in some cases provide too much flexibility. This issue can be tackled by defining a notion of quality over assumption-based plans. We discuss this in more detail in the preferred assumption-based planning section.

Initial-State Assumption-Based Planning

In many settings it is convenient or sufficient to make assumptions only about the initial state of the world. In other words, to make $h_i = \mathtt{true}$ for every $i > 0$. We call this class of problems initial-state assumption-based planning. An initial-state assumption-based plan is denoted by (h_0, α), where h_0 is a boolean formula over T that corresponds to an assumption made on the initial state.

The formal relation between conformant planning and initial-state assumption-based planning is straightforward, and is established in the following proposition.

Proposition 1 *The tuple* (h_0, α) *is an initial-state assumption-based plan for planning problem* $P = (F, O, I, G, T)$ *iff* α *is a conformant plan for* $P' = (F, O, I \cup h_0, G)$.

Note that this proposition *does not* imply that an assumption-based plan can be directly computed using a conformant planner, since a conformant planner is not able to compute assumptions. In addition, a relation between assumption-based planning and initial-state assumption-based planning can be established.

Theorem 1 *If* $P = (F, O, I, G, T)$ *and* (ρ, α) *is an assumption-based plan for* P*, then there exists an* h_0 *such that* (h_0, α) *is an initial-state assumption-based plan for* P*. Furthermore,* h_0 *can be computed from* ρ*,* P *and* α *in time* $2^{\mathcal{O}(|\alpha|)}$.

As a consequence of this theorem, if α is a sequence of actions for which there is some ρ such that (ρ, α) is an assumption-based plan, then we can construct an assumption h_0 on the initial state using α such that (h_0, α) is an assumption-based plan. The proof of the above theorem (omitted here for space) actually gives a constructive algorithm for h_0 that relies on regressing G over α. We now analyze aspects that relate to the complexity of assumption-based planning. As it turns out, the definition of assumption-based planning is general enough that its complexity seems to lie across a spectrum of complexity classes, depending on which literals are allowed to be assumed. Below we provide two complexity results showing that assumption-based planning is complete for two complexity classes. Our first result follows directly from the fact that conformant planning is EXPSPACE-complete (Haslum and Jonsson 1999).

Theorem 2 *Given an assumption-based planning problem* $P = (F, O, I, G, T)$*, where* T *contains no fluents mentioned in non-unary clauses of* I*, deciding whether or not an assumption-based plan exists is EXPSPACE-complete.*

However, as more information can be assumed, the complexity of the decision problem move down to that of classical planning.

Theorem 3 *Given an assumption-based planning problem* $P = (F, O, I, G, T)$*, where* T *contains all fluents mentioned in non-unary clauses of* I*, deciding whether or not an assumption-based plan exists is PSPACE-complete.*

Naive Approach to Assumption-Based Planning

Theorem 3 implies that when the set of assumable fluents contain all fluents appearing in non-unary clauses of I, assumption-based planning can be reduced to classical planning. Indeed, a naive algorithm for this type of assumption-based planning can be proposed by building a classical planning problem in which the planner first has to "guess" an assumption on the initial state, and then find a sequence of actions.

Specifically, the classical problem P' is like P but augmented with additional actions that can only be performed at the initial state and have as an objective to generate an initial state consistent with I. There is an exponential number of these actions. If $a_0 a_1 \ldots a_n$ is a plan for P, we construct the initial-state assumption-based plan α as follows. h_0 is constructed with the facts true in the state s that a_0 generates. a is simply set to $a_1 \ldots a_n$. Of course, the approach derived by the proof of this theorem is very impractical as it grows the size of the problem exponentially. In an extended version of our car example, in which we have n components of the car whose state is unknown, we would have 2^n actions that complete the initial state. Alternatively, some domains may lend themselves to achieve the same completion effect by applying a sequence of actions. In our example, each sequence of these actions generates one of the possible 2^n states.

An important limitation of both of the aforementioned approaches is that actions performed at the beginning of the plan make an explicit commitment to a *single* initial state. In many practical applications, such a compromise seems too excessive. Computationally, committing to a single state may lead the search astray. From a high-level perspective, committing to a single state produces assumptions that may be too restrictive, which may be undesirable. Both approaches outlined above have been used in the past to tackle diagnosis problems in which the initial state is unknown (Sohrabi, Baier, and McIlraith 2010; Haslum and Grastien 2011).

A Translation-Based Approach

In this section we propose an alternative translation of assumption-based planning into classical planning that builds on top of Palacios and Geffner's $K_{T,M}$ translation (2009) – henceforth denoted by P&G– which translates conformant planning into classical planning. The main objective of our translation is to avoid the excessive commitment exhibited by the naive translation of assumption-based planning into classical planning. We describe the basics of the translation, analyze its properties, and finally compare it to other extensions of P&G.

The $K^A_{T,M}$ Translation

Given a planning problem $P = (F, O, I, G)$, we generate a new planning problem $P' = (F', O', I', G')$; we call this process the $K^A_{T,M}$ translation, which builds on P&G. For each literal L we associate a set of *merges*, M_L. Each merge is a finite set of *tags*, which in turn are conjunctions of literals that are unknown in the initial state. Each merge characterizes a partition of the initial state in the sense that $I \models \bigvee_{t \in m} t$ is required to hold for each merge m.

A tag intuitively represents a partial completion of the initial state in which every $L \in t$ is initially true – it is a "case" in which L is initially true. Problem P' contains fluents of the form KL, for each $L \in F$, Kt and $K\neg t$ for each tag t, and KL/t for each $L \in F$ and each tag t in a merge of M_L. KL intuitively represents that L is *known*. KL/t represents the fact that L is known given that t is true in the initial state.

The main difference between P&G and our translation is that we consider a set of *assumption actions*, which allow the planner to assume that a tag t was true in the initial state. More specifically, given a set $\mathcal{T}$ of *assumable tags*, there is an assumption action associated to each $t \in \mathcal{T}$ that assumes t is true in the initial state. Instead of using the standard P&G merge actions, we use the contingent merge actions introduced by Albore, Palacios, and Geffner (2009). Furthermore, our translation augments P&G with additional conditional effects to handle assumptions. More precisely, P' is such that the following holds.

(1) $I' = \{KL \mid I \models L\} \cup \{KL/t \mid I, t \models L\} \cup \{ok\}$. I' differs from P&G only in the ok fluent which is added to keep track of consistency and is explained in detail later.

(2) For each literal L, and each tag t in some merge of M_L, O contains the so-called *contingent merges* proposed originally by Albore, Palacios, and Geffner (2009), of the form $[\bigwedge_{t \in m}(KL/t \vee K\neg t)] \rightarrow KL$. These generalize the P&G merge actions for the case where t is refuted by assumptions.

(3) Like in P&G, for each action a with conditional effect $C \rightarrow L$, O' contains the conditional effects $[\bigwedge_{c \in C} Kc/t] \rightarrow KL/t$ and $[\bigwedge_{c \in C} \neg K\neg c/t] \rightarrow \neg K\neg L/t$, for each tag t in some merge of M_L.

(4) In addition, for each tag t in the set of assumable tags, $\mathcal{T}$, we create an assumption action $Assume(t)$, with precondition $\neg K\neg t \wedge \neg Kt \wedge \neg K\neg t' \wedge \neg Kt'$ and effects Kt, $\neg K\neg t$, $K\neg t'$, $\neg Kt'$ for every tag t' that is inconsistent with t, i.e., contains the complement of a literal in t.

(5) For each merge set M_L that contains tag t, and each merge $m \in M_L$, the conditional effects $KL/t \rightarrow KL$, and $KL/t \wedge K\neg L \rightarrow \neg ok$ are added to the $Assume(t)$ action. The first conditional effect makes L known if it is the case that KL/t. The second conditional effect takes care of potential inconsistencies that could arise when assuming a literal that implies that L is known, when $\neg L$ is already known. In such cases the action deletes the fluent ok signaling inconsistency.

(6) For each action $a \in O$ the version of a in O' contains the precondition $ok \wedge \bigwedge_{L \in prec(a)} KL$.

(7) The planner should not make inconsistent assumptions. Thus whenever we assume a tag t, we may need to update the knowledge about other tags. We achieve this by adding specific conditional effects to assumption actions. Such effects reflect logical inferences among clauses defining the initial state and we obtain them by performing resolution. For the sake of space, we refer the reader to the unabridged paper for a detailed explanation of the resolution step.

(8) Finally, $G' = \{KL \mid L \in G\} \cup \{ok\}$.

Just like P&G, our $K^A_{T,M}$ translation is *sound* in the following sense.

Theorem 4 *The $K^A_{T,M}$ translation is sound; i.e., if α is a plan for $K^A_{T,M}(P)$, then there is an assumption-based plan (ρ, α') for the original problem. Furthermore, (ρ, α') can be computed from α in linear time.*

The $K^A_i(P)$ Translation

As with P&G's $K_{T,M}$ translation, the $K^A_{T,M}$ translation does not define explicitly how the merges/tags are computed from the original problem. In addition, it provides no completeness guarantees. A practical realization of P&G's $K_{T,M}$ is given by the so-called K_i translation (Palacios and Geffner 2009). K_i defines an explicit way to compute merges. It is a sound translation (in the sense defined above). In addition, if i is not greater than the so-called *width* of the problem P, then it is also complete.

We have defined an analogous version of the K_i translation, that we call K^A_i. K^A_i is a version of K_i in which merges and tags are computed using the same procedure as for the case of K_i. Due to lack of space we cannot elaborate on this process, but we refer the reader to Palacios and Geffner's paper (2009) for reference. After the tags and merges are determined, however, it may be that the set of tags does not capture the set of assumable fluents. In such a case, we create additional tags for those assumable fluents that are not captured.

Since K^A_i is a particular form of the translation $K_{T,M}$, we obtain that it is *sound* as a corollary of Theorem 4. Furthermore,

Theorem 5 *Given an assumption-based planning problem $P = (F, O, I, G, T)$, with width $w(P) \leq i$, the K_i^A translation is complete; i.e. if there exists an assumption-based plan (ρ, α) for P, in which ρ are conjunctions of literals in T, then a plan exists for $K_i^A(P)$.*

In the previous result, $w(P)$ is defined analogously to P&G.

Negative Results Given P, $K_i(P)$ is polynomial in the width of P (Palacios and Geffner 2009). Since our implementation involves a step in which previously we do a resolution fixpoint computation (Step (7)) we cannot guarantee that the K_i^A translation is polynomial on the width of P.

Preferred Assumption-Based Planning

The definition of an assumption-based plan allows the planner to assume any aspect of the state that can be constructed from the subset of assumable literals and consistently assumed. However, some assumptions will be more reasonable than others. E.g., in our car example, if it's summer, it may be much more reasonable to assume $batteryOk$ than that the car $hasGas$. In the cold of winter, the opposite may be true. To define the notion of a preferred assumption-based plan, we employ a preference relation $\preceq$, a transitive and reflexive relation in $\Pi \times \Pi$, where Π contains precisely all assumption-based plans for a particular planning instance (following Baier and McIlraith 2008). Plan optimality is defined in the obvious way given relation $\preceq$.

For the purposes of this paper, we will appeal to the uniform notion of action cost in order to characterize preferred assumption-based plans, rather than defining $\preceq$ directly. Specifically, given an assumption-based planning problem P, we build its translated instance $K_{T,M}^A(P) = (F, O, I, G)$, and then augment this instance to produce a *cost-based planning problem* $P_{\mathcal{C}} = (F, O, I, G)$ such that each action $a \in O$ has a non-negative cost $\mathcal{C}(a)$. Note that this means that actions of the form $Assume(t)$ will have a cost associated with them. Likewise, so do the domain actions and merge actions. The task reduces now to finding a cost-optimal plan.

Specifying how a domain expert would specify these preferences in the original problem specification and ensuring that the corresponding cost-based planning problem respects the induced $\prec$ relation can be achieved in a variety of ways. Detailed discussion of this issue is beyond the scope of this paper. For the purposes of illustrating some of the properties of this approach, we can directly and intuitively add costs to the translated classical planning problem, as we do in the section to follow.

Implementation and Experiments

The K_1^A translation was implemented in our A0 planner as an augmentation of Palacios and Geffner's T0 planner. In the absence of the specification of assumables, the assumables are set to all the fluents less those involved in G, precluding assumption-based plans that assume G. We use FF (Hoffmann and Nebel 2001) to generate classical plans with the translated domains and convert them back into assumption-based plans. To generate preferred assumption-based plans, we associate a cost with each action in the (translated) classical planning problem. The resulting cost-based planning problem is solved using Metric-FF and LAMA.

Since the notion of assumption-based planning is new, there are no systems to benchmark against. We sought instead to evaluate the running time of A0 + FF compared to an implementation of so-called *naive* assumption-based planning, and to various cost distributions for cost-based assumption-based planning. We also sought to assess gross properties of the translation: proportion of solution time; and size relative to its T0 counter part, and to the original problem.

Domains: We exploited four domains from the International Planning Competition (IPC) benchmark suite: *logistics*, *raokeys*, *coins*, and *blocks*. Experiments are still in progress for the latter two domains, but initial results are promising; details will be given in the full paper. We augmented the logistics domain by adding gas levels to trucks that decremented, gas stations and refueling actions, and we modified some intracity connections to be uni-directional or missing. We refer to this modified domain as *alogistics*. The 12 instances we constructed varied in the number of cities and trucks (2-4), and locations within a city. Varying amounts of uncertainty were introduced into the initial state of each instance via unknown truck gas levels and locations, and the connectedness within cities.

The second domain we used was *raokeys*, a conformant planning benchmark from IPC-2008. The problem requires reasoning about n locks with n different possible keys in n different possible locations, making the number of initial states combinatorial explosive. Conformant plans for this domain are long and must consider a combinatorially explosive number of possible initial states. In contrast the problem is simple for a human. This makes it a challenging problem for assumption-based planning. We experimented with 4 instances of this conformant domain, for $n = 2, \ldots, 5$.

Experiments: We ran 7 different experimental configurations on the 16 problem instances described above and 4 preliminary results from *coins*. **(1)** We ran A0 + FF on the translated domains. **(2)** We generated a naive assumption-based planning problem by augmenting each problem instance with actions that create each of the different consistent completions of the initial state, then solved with FF. **(3)–(7)** These configurations all relate to generating preferred assumption-based plans. The configurations differ with respect to the cost of the assumption actions relative to the domain actions. E.g., $x = 0.5$ denotes that assumption actions are twice as expensive as domain actions. All merge actions were assigned equal (low) cost. Instances solved via A0 + Metric-FF or LAMA.

Table 1 shows the results obtained on the seven configurations for 20 instances. On the classical settings, A0 does not take much more time than the naive method but makes far fewer assumptions. Problems alog-1 to alog-4, are all variants of the same problem instance but with progressively more unimportant uncertainty. This domain is contrasted by the *raokeys* problem instances in which A0 seems to take ex-

Table 1: Comparing the seven configurations with the total time to solve in seconds on the left and plan length on the right. The number of assumptions made appears in parentheses. The results are preliminary and experiments are in progress (X: unknown failure in back-end planner. TTO: time out during translation. STO: time out during solving classical plan). All experiments were run on a 2.80GHz machine with 2GB memory and a 30 minute timeout.

Prob	**Classical** (t(s)/len)		**Cost-Based Metric-FF** (t(s)/len)					**Cost-Based LAMA** (t(s)/len)				
	A0	naive	x=0.1	x=0.5	x=1	x=2	x=10	x=0.1	x=0.5	x=1	x=2	x=10
alog-1	0.06/44(2)	0.00/39(4)	0.26/39(2)	0.04/39(2)	0.03/39(2)	0.05/39(3)	0.29/39(3)	101.75/36(1)	37.58/36(1)	0.03/38(2)	0.02/38(2)	0.02/38(2)
alog-2	0.07/44(2)	0.01/ 51(16)	0.33/39(2)	0.04/39(2)	0.04/39(2)	0.06/39(3)	0.36/39(3)	6.03/36(1)	0.03/38(2)	0.02/38(2)	220.53/36(2)	251.66/36(2)
alog-3	0.06/44(2)	0.02/ 61(26)	0.33/39(2)	0.03/39(2)	0.04/39(2)	0.06/39(3)	0.35/39(3)	6.03/36(1)	0.03/38(2)	0.03/38(2)	0.03/38(2)	253.99/36(2)
alog-4	0.11/44(2)	0.27/ 129(94)	0.41/39(2)	0.05/39(2)	0.05/39(2)	0.06/39(3)	0.43/39(3)	6.35/36(1)	475.12/36(1)	0.04/41(2)	589.5/36(2)	0.05/41(2)
alog-5	0.06/50(2)	0.01/ 47(5)	0.39/45(2)	0.05/45(2)	0.05/45(2)	0.09/45(3)	0.69/45(3)	54.14/42(1)	38.67/44(2)	38.71/44(2)	38.53/44(2)	38.04/44(2)
alog-6	0.07/50(2)	0.01/ 50(8)	0.41/45(2)	0.06/45(2)	0.05/45(2)	0.11/45(4)	0.77/45(4)	4.45/42(1)	1.41/43(2)	1.49/43(2)	1.41/43(2)	1.45/43(2)
alog-7	0.07/52(3)	0.01/ 54(5)	0.03/51(3)	0.03/51(3)	0.03/51(3)	0.03/51(3)	0.07/51(3)	0.65/54(3)	0.67/54(3)	0.64/54(3)	0.66/54(3)	0.65/54(3)
alog-8	0.1/54(3)	0.01/ 50(6)	0.04/53(2)	0.03/53(2)	0.04/53(3)	0.05/52(4)	2.45/51(3)	13.6/55(2)	13.25/55(2)	13.32/55(2)	524.5/49(3)	669.47/51(3)
alog-9	0.08/56(5)	0.01/ 51(10)	0.10/61(2)	0.16/61(2)	0.15/61(4)	0.08/57(6)	9.80/51(4)	427.26/60(2)	423.53/52(3)	424.44/52(3)	423.99/52 93)	424.19/52(3)
alog-10	0.06/39(5)	0.01/ 36(6)	0.36/37(4)	0.04/39(5)	0.03/39(6)	0.06/36(5)	0.50/36(5)	8.93/33(1)	12.52/34(1)	0.03/37(4)	0.02/37(4)	156.15/33(4)
alog-11	0.14/55(6)	0.01/ 50(14)	1671.78/46(7)	0.21/48(9)	0.23/48(9)	1.56/52(11)	975.88/46(9)	53.21/43(1)	7.29/50(1)	71.76/51(6)	3.49/50(11)	99.78/49(11)
alog-12	0.16/61(8)	0.04/72(17)	160.64/61(5)	0.25/61(5)	0.32/58(4)	2.75/56(5)	STO	X	X	X	X	X
rao-2	0.05/10(2)	0.01/ 10(2)	0.02/ 10(2)	0.02/10(2)	0.03/ 10(2)	0.12/10(2)	33.33/10(2)	0.02/10(2)	0.02/10(2)	0.02/10(2)	0.02/10(2)	0.02/10(2)
rao-3	36.39/16(3)	0.04/ 24(14)	X	X	X	X	X	60.79/17(3)	57.93/18(3)	61.75/17(3)	75.57/17(4)	80.81/17(4)
rao-4	X	3.96/ 36	X	X	X	X	X	X	X	X	X	X
rao-5	X	X	X	X	X	X	X	X	X	X	X	X
coins-1	0.02/8(3)	0.01/10(4)	0.01/8(3)	0.01/8(3)	X	X	X	0.02/9(0)	X	0.02/8(3)	0.02/8(3)	0.02/8(3)
coins-7	0.05/13(5)	0.01/15(6)	0.03/13(5)	0.03/13(5)	0.04/13(5)	137.67/13(5)	STO	9.1/26(0)	0.12/13(4)	0.07/13(5)	0.07/13(5)	0.04/13(5)
coins-8	0.08/14(5)	0.01/16(6)	0.04/14(5)	0.03/14(5)	0.04/14(5)	137.86/14(5)	STO	10.07/26(0)	0.29/14(4)	0.07/14(5)	0.07/14(5)	0.04/14(5)
coins-17	0.61/21(7)	0.09/24(10)	X	X	X	X	X	137.61/22(6)	4.85/21(7)	3.71/21(7)	2.79/21(7)	2.93/21(7)
coins-29	X	31.51/70(33)	X	X	X	X	X	129.55/70(20)	149.45/68(20)	137.65/70(20)	132.58/69(20)	146.67/70(20)

ponentially more time while the naive method works fairly well. Strong conclusions cannot be made however because of the small number of instances that can be solved.

The cost-based run on metric-FF showed a clear trend that $x = 0.1$ and $x = 10$ took longer to solve than values of x closer to 1. All the *alogistic* instances require both assumptions and regular actions to be solved. When the cost discrepancies are high, the search may be too focused building plan prefixes with the cheapest action, without considering other (useful) actions.

We evaluated the size of our A0 translations relative to the original problem and to a comparable T0 translation. Given the diversity of domains and the small number, our observations are somewhat anecdotal. For the *alogistics* domains, A0 and T0 performed reasonably consistently. The number of atoms was approximately double that in the original domain and there was a very small increase in the number of actions. In the *raokeys* domains A0 and T0 both doubled the number of actions, but there was no other clear trend, except that A0 had one or two orders of magnitude more conditional effects than the original domain, whereas T0 only had a constant factor increase. These large number of conditional effects are due to the dependencies between clauses in the domain which results in a high number of clauses that can be produced by the resolution algorithm.

We also evaluated the proportion of solution time dedicated to the translation. For the *alogistics* domains this ranged from 5 - 25%, whereas with the *raokeys* domain, it was closer to 50%. On the other hand, A0 took about $0.1 - 0.6$ times the amount of time to translate *alogistics* problem instances as T0. On raokeys2 the time spent was about the same, but on raokeys3 A0 took almost 100 times as long. Again, this is due to the unit propagation required for the assumption actions.

Summary and Concluding Remarks

In this paper we introduce the notion of assumption-based planning. We provide a formal characterization of assumption-based planning, establishing a correspondence to conformant planning. Exploiting this correspondence, we provide a translation of an assumption-based planning problem to a classical planning problem, building on the popular translation developed by P&G. We prove the soundness and completeness of our translation. This provides us with a means of generating assumption-based plans using classical planners. We also argue for the merit of preferred assumption-based plans and propose a means of realizing such plans using cost-based planning. We describe A0, a planner that addresses the subset of initial state assumption-based planning problems and present experiments that illustrate the viability of our approach and that assess some properties of our translation.

While this paper explores the generation of assumption-based plans via a translation to classical planning, the correpondence to conformant planning opens the door to adapting a variety of conformant planners for assumption-based planning (e.g., To, Son, and Pontelli 2010). Beyond planning, the assumption-based planning paradigm has compelling applications in diagnosis and verification of dynamical systems.

Acknowledgements: We gratefully acknowledge funding from the Natural Sciences and Engineering Research Council of Canada (NSERC). Jorge Baier was partly funded by Fondecyt grant no. 11110321.

References

Albore, A., and Bertoli, P. 2004. Generating safe assumption-based plans for partially observable, nondeterministic domains. In *Proc. of the 19th National Conference on Artificial Intelligence (AAAI)*, 495–500.

Albore, A., and Bertoli, P. 2006. Safe ltl assumption-based planning. In *Proc. of the 16th International Conference on Automated Planning and Scheduling (ICAPS)*, 193–202.

Albore, A., and Geffner, H. 2009. Acting in partially observable environments when achievement of the goal cannot be guaranteed. In *Proc. of ICAPS Workshop on Planning and Plan Execution for Real-World Systems*.

Albore, A.; Palacios, H.; and Geffner, H. 2009. A translation-based approach to contingent planning. In *Proc. of the 21st International Joint Conference on Artificial Intelligence (IJCAI)*, 1623–1628.

Bacchus, F., and Kabanza, F. 2000. Using temporal logics to express search control knowledge for planning. *Artificial Intelligence* 116(1-2):123–191.

Baier, J. A., and McIlraith, S. A. 2008. Planning with preferences. *Artificial Intelligence Magazine* 29(4):25–36.

Bonet, B., and Geffner, H. 2011. Planning under partial observability by classical replanning: Theory and experiments. In *Proc. of the 22nd International Joint Conference on Artificial Intelligence (IJCAI)*.

Conrad, P. R., and Williams, B. C. 2011. Drake: An efficient executive for temporal plans with choice. *Journal of Artificial Intelligence Research* 42:607–659.

Göbelbecker, M.; Gretton, C.; and Dearden, R. 2011. A switching planner for combined task and observation planning. In *Proc. of the 26th AAAI Conference on Artificial Intelligence (AAAI)*.

Haslum, P., and Grastien, A. 2011. Diagnosis as planning: Two case studies. In *Proc. of the International Scheduling and Planning Applications workshop (SPARK)*.

Haslum, P., and Jonsson, P. 1999. Some results on the complexity of planning with incomplete information. In *Proc. of the 5th European Conference on Planning (ECP)*, 308–318.

Hoffmann, J., and Nebel, B. 2001. The FF planning system: Fast plan generation through heuristic search. *Journal of Artificial Intelligence Research* 14:253–302.

Palacios, H., and Geffner, H. 2009. Compiling uncertainty away in conformant planning problems with bounded width. *Journal of Artificial Intelligence Research* 35:623–675.

Pellier, D., and Fiorino, H. 2004. Assumption-based planning. In *In Proceedings of the International Conference on Advances in Intelligence Systems Theory and Applications, Luxemburg*.

Pellier, D., and Fiorino, H. 2005. Multi-agent assumption-based planning. In *Proc. of the 19th International Joint Conference on Artificial Intelligence (IJCAI)*, 1717–1718.

Poole, D.; Goebel, R.; and Aleliunas, R. 1987. Theorist: a logical reasoning system for defaults and diagnosis. In Cercone, N., and McCalla, G., eds., *The Knowledge Frontier: Essays in the Representation of Knowledge*. New York: Springer Verlag. 331–352.

Reiter, R., and de Kleer, J. 1987. Foundations of assumption-based truth maintenance systems: Preliminary report. In *Proc. of the 6th National Conference on Artificial Intelligence (AAAI)*, 183–189.

Sohrabi, S.; Baier, J.; and McIlraith, S. A. 2010. Diagnosis as planning revisited. In *Proc. of the 12th International Conference on Knowledge Representation and Reasoning (KR)*, 26–36.

Sohrabi, S.; Baier, J. A.; and McIlraith, S. A. 2011. Preferred explanations: Theory and generation via planning. In *Proc. of the 26th AAAI Conference on Artificial Intelligence (AAAI)*, 261–267.

To, S. T.; Son, T. C.; and Pontelli, E. 2010. A new approach to conformant planning using cnf*. In *Proc. of the 20th International Conference on Automated Planning and Scheduling (ICAPS)*, 169–176.