From Conformant into Classical Planning: Efficient Translations That May be Complete Too

Héctor Palacios
Departamento de Tecnología
Universitat Pompeu Fabra
08003 Barcelona, SPAIN
`hector.palacios@upf.edu`

Héctor Geffner
Departamento de Tecnología
ICREA & Universitat Pompeu Fabra
08003 Barcelona, SPAIN
`hector.geffner@upf.edu`

Abstract

Focusing on the computation of conformant plans whose verification can be done efficiently, we have recently proposed a polynomial scheme for mapping conformant problems P with deterministic actions into classical problems $K(P)$. The scheme is sound as the classical plans are all conformant, but is incomplete as the converse relation does not always hold. In this paper, we extend this work and consider an alternative, more powerful translation based on the introduction of epistemic tagged literals KL/t where L is a literal in P and t is a *set* of literals in P unknown in the initial situation. The translation ensures that a plan makes KL/t true only when the plan makes L certain in P given the assumption that t *is initially true*. We show that under general conditions the new translation scheme is complete and that its complexity can be characterized in terms of a parameter of the problem that we call *conformant width*. We show that the complexity of the translation is exponential in the problem width only, find that the width of almost all benchmarks is 1, and show that a conformant planner based on this translation solves some interesting domains that cannot be solved by other planners. This translation is the basis for T_0, the best performing planner in the Conformant Track of the 2006 International Planning Competition.

Introduction

Conformant planning is the problem of finding a sequence of actions for achieving a goal in the presence of uncertainty in the actions or initial state (Goldman & Boddy 1996; Smith & Weld 1998). The problem is more complex than classical planning, as the verification of plans itself is intractable in the worst case (it must consider all possible initial states and transitions) but simpler than the more general problem of planning with partial observability (Haslum & Jonsson 1999; Rintanen 2004). While few practical problems are purely conformant, the ability to find conformant plans appears to be a necessity in contingent planning where conformant situations are an special case (null observability being an special case of partial observability) and where relaxations into conformant planning appear to provide useful heuristics (Brafman & Hoffmann 2004).

In (Palacios & Geffner 2006), we have recently proposed a polynomial scheme for mapping conformant problems P

with deterministic actions into classical problems $K(P)$ that can then be solved efficiently by an off-the-shelf classical planner. The scheme is sound in sense that the classical plans obtained from $K(P)$ are all conformant, but is incomplete, as $K(P)$ may admit no solution when P does. As shown empirically in (Palacios & Geffner 2006), however, few of the existing benchmark domains cannot be solved in this way, and those that can be solved in this way, are normally solved faster than by other methods. This is apparent from the results of the recent Conformant Planning Competition held at ICAPS-06, available from (Bonet & Givan 2006), where the translation $K(P)$ was fed to the classical FF planner (Hoffmann & Nebel 2001) producing a planner, KP, that was dominated only by T_0, the planner based on the translation that is developed in this paper.

The $K(P)$ translation is defined with the aid of new literals KL and L/X for literals L and X in the problem P, ensuring that KL and L/X are true when L and the conditional 'if X then L' are known to be true respectively. The tagged literals L/X are 'produced' by conditional effects of the form $a : C \wedge X \rightarrow L$, translated into $a : KC \rightarrow L/X$, and are 'consumed' by merge actions of the form $d : L/X_1, L/X_2 \rightarrow KL$ provided that $X_1 \vee X_2$ is known to hold, accounting thus for a simple form of reasoning by cases.

In this paper, we extend this work and consider an alternative translation scheme $K_i(P)$, where i is a non-negative integer, that overcomes some of the theoretical and practical limitations of $K(P)$. In particular, we define a parameter $w(P)$ that we call the *conformant width* of P, and show that the translation $K_i(P)$ is provably complete when $w(P) \leq i$, while being exponential only in i. Moreover, we will see that almost all conformant benchmarks have actually width equal to 1. The planner T_0 takes advantage of this fact and is built on the translation $K_i(P)$ for $i = 1$.

The translation $K_i(P)$ is stronger than $K(P)$ for any $i \geq 1$ while for $i = 0$ coincides with the *core* translation $K_0(P)$ (Palacios & Geffner 2006) that captures the plans valid under the so-called 0-approximation (Baral & Son 1997). The key departures from $K(P)$ are in the syntax and semantics of the conditionals represented by the 'tagged' literals KL/t: syntactically, t no longer has to be a single literal but can be an arbitrary set of literals, and semantically, the truth of KL/t no longer means that L is true if t is true

at the same time, but that L is true if t was true *in the initial situation*. In other words, the tags t will refer to possible local contexts in the initial situation, and the tractability results for problems P with bounded width $w(P)$ mean that only a bounded number of such local contexts need to be considered.

Conformant planning can be formulated as the search for a sequence of applicable actions that map an initial belief state b_0 into a target belief state b_F. A belief state bel represents the set of states s that are deemed possible, and actions a, whether deterministic or not, deterministically map one belief state bel into another, denoted as bel_a. This view is formulated in (Bonet & Geffner 2000) where an admissible heuristic is introduced for solving this shortest-path problem over belief space. More recent proposals, incorporate more effective belief state representations and heuristics (Hoffmann & Brafman 2005; Cimatti, Roveri, & Bertoli 2004; Bryce, Kambhampati, & Smith 2006), while others build directly on incomplete but tractable belief representations (Son *et al.* 2005). The translation-based approach combines and extends these ideas providing *a tractable but incomplete belief representation* that at the same time benefits from the *heuristics* that have been proved successful in classical planning for guiding the search for plans in such a simplified belief space (Palacios & Geffner 2006). The approach is also closely related to the notion *planning at the knowledge level* formulated in (Petrick & Bacchus 2002) except that the epistemic encoding is derived automatically and is solved by an off-the-shelf classical planner that uses an automatically derived heuristic as well.

The paper is organized as follows. We consider first the translation $K(P)$ and some of its limitations. Then we introduce an alternative translation scheme $K_{T,M}$ that is based on two parameters: a set of tags t and a set of merges m. We consider next two special instances of this general scheme: the $K_{S0}(P)$ translation that is complete but exhaustive as T is selected to stand for the set of all possible initial states, and the $K_i(P)$ translation that is complete for problems with width $w(P) \leq i$ and involves tags of size no greater than i. The translation $K_i(P)$ for $i = 1$ is then used as the basis of a conformant planner that is tested over many domains.

The translation $K(P)$

Following (Palacios & Geffner 2006), a conformant planning problem P is a tuple $P = \langle F, O, I, G \rangle$ where F stands for the fluent symbols in the problem, O stands for a set of actions a, I is a set of clauses over F defining the initial situation, and G is a set of literals over F defining the goal. In addition, every action a has a precondition given by a set of fluent literals, and a set of conditional effects $C \to L$ where C is a set of fluent literals and L is a literal. All actions are assumed to be *deterministic* and hence all uncertainty lies in the initial situation.

We refer to the conditional effects $C \to L$ of an action a as the *rules* associated with a, and sometimes write them as $a : C \to L$. Also, we use the expression $C \wedge X \to L$ to refer to rules with literal X in their bodies. In both cases, C may be empty. Last, when L is a literal, we take $\neg L$ to denote the complement of L.

The basic core of the translation $K(P)$ in (Palacios & Geffner 2006) maps P into a *classical planning problem* $K_0(P) = \langle F', O', I', G' \rangle$ where[1]

- $F' = \{KL, K\neg L \mid L \in F\}$
- $I' = \{KL \mid L \in I\}$
- $G' = \{KL \mid L \in G\}$
- $O' = O$ but with each literal precondition L for $a \in O$ replaced by KL, and each conditional effect $a : C \to L$ replaced by $a : KC \to KL$ and $a : \neg K\neg C \to \neg K\neg L$.

For any literal L in P, KL denotes its 'epistemic' counterpart in $K_0(P)$ whose meaning is that L is known. The expressions KC and $\neg K\neg C$ for $C = L_1 \wedge L_2 \ldots$ are used as abbreviation of the formulas $KL_1 \wedge KL_2 \ldots$, and $\neg K\neg L_1 \wedge \neg K\neg L_2 \ldots$.

The intuition behind the translation is simple: first, the literal KL is true in the initial state I' if L is known to be true in I; else it is false. This removes all uncertainty from $K_0(P)$. In addition, to ensure soundness, each conditional effect $a : C \to L$ in P maps, not only into the **supporting rule** $a : KC \to KL$ but also into the **cancellation rule** $a : \neg K\neg C \to \neg K\neg L$ that guarantees that $K\neg L$ is deleted (prevented to persist) when action a is applied when C is not known to be false.

The core translation $K_0(P)$ is sound as every classical plan that solves $K_0(P)$ is a conformant plan for P, but it is incomplete, as not all conformant plans for P are classical plans for $K(P)$. The meaning of the KL literals follows a similar pattern: if a plan achieves KL in $K_0(P)$, then the same plan achieves L with certainty in P, yet a plan may achieve L with certainty in P without making the literal KL true in $K_0(P)$.

The translation $K(P)$ extends $K_0(P)$ with new literals, actions, and effects, enabling the solution of a wider range of problems by accounting for a simple form of reasoning by cases. The new literals L/X_i where L is a literal in P and X_i is a literal in a clause $X_1 \vee \cdots \vee X_n$ in I, stand for the conditionals 'if X_i then L' as long as the boolean flag $F_{X,L}$, initially set to true, remains true. The new translation rules added in $K(P)$ are (see (Palacios & Geffner 2006) for details):

- **Split:** If $a : C \wedge X_i \to L$ and $X_1 \vee \cdots \vee X_n$ in P for some $i \in [1, n]$, then add $a : KC \to L/X_i$ to $K(P)$, initializing the literal L/X_i to **false**

- **Merge:** If $X_1 \vee \cdots \vee X_n$ in P, then add new action $merge_{X,L}$ to $K(P)$ with conditional effect

  $$(L/X_1 \vee K\neg X_1) \wedge \cdots \wedge (L/X_n \vee K\neg X_n) \wedge F_{X,L} \to KL$$

- **Protect:** If $a : C \to Y$ in P, then add $a : \neg K\neg C \to \neg F_{X,L}$ in $K(P)$ when Y is $\neg L$ or $\neg X_i$.

Conditionals L/X_i are thus 'produced' when a conditional effect $C \wedge X_i \to L$ is triggered and C is known (**Split**) and are 'consumed' producing KL when all L/X_i hold along with the disjunction $X_1 \vee \cdots \vee X_n$ holds (**Merge**). The

---

[1] For every rule $a : KC \to KL$ introduced in $K(P)$, the rule $a : KC \to \neg K\neg L$ is also added that enforces the invariant $\neg(KL \wedge K\neg L)$.

persistence of the clause and the conditionals is represented by a boolean flag $F_{X,L}$ which is initially true but is deleted when effects that can delete either L or some X_i cannot be ruled out with certainty (**Protect**). The mapping $K(P)$ also includes the translation rule:

- **Action Compilation:** If $a : C \wedge \neg L \to L$ in P, then add
  $a : KC \to KL$ to $K(P)$

that yields the head L of a conditional effect with certainty when all body literals are known with the exception of $\neg L$.

The range of conformant problems P that can be solved with the translation $K(P)$ is illustrated by the results in (Palacios & Geffner 2006) and those of the recent conformant planning competition (Bonet & Givan 2006) where the output of the translation $K(P)$ was fed into the classical FF planner (Hoffmann & Nebel 2001) with good results.

The classical plans for $K(P)$ are valid conformant plans for P once the merge actions are removed. On the other hand, P may have valid conformant plans that are lost in $K(P)$. This is shown below.

Example 1 Consider the problem of moving an object from an origin to a destination using two actions: $pick(l)$ that picks up an object from a location if the object is in that location, and $drop(l)$ that drops the object in a location if the object is being held. Let us assume that these are conditional effects, and that there are no preconditions. Given an instance P where the object is at either l_1 or l_2, and must be moved to l_3, $K(P)$ has a solution

$$\pi_1 = \{pick(l_1), pick(l_2), merge_{X,L}, drop(l_3)\}$$

where $X = l_1 \vee l_2$ and $L = hold$, as the first action yields $hold/at_1$, the second $hold/at_2$, and the third $Khold$. This plan with the merge action removed is a conformant plan for P.

Let us now modify the problem slightly so that $pick(l)$ picks up the object if the object is at l *and* the hand is empty, while if the hand is not empty, $pick(l)$ just releases the object at l. For this variation the plan π_1 above does not work for either $K(P)$ or P because if the first action $pick(l_1)$ is successful then the second action $pick(l_2)$ cannot be. Actually, $K(P)$ has no solution then, and yet P has the conformant solution

$$\pi_2 = \{pick(l_1), drop(l_3), pick(l_2), drop(l_3)\}$$

By analyzing the beliefs that result from this plan, one can see why this plan cannot be captured in $K(P)$: the actions $pick(l_i)$ produce the conditionals $hold/at(l_i)$, yet these conditionals cannot be merged to yield $Khold$ as indeed $hold$ is never true with certainty along the execution of π_2. We will show next, however, that such plans can be captured by a similar translation where conditionals have a slightly different meaning.

General Translation Scheme $K_{T,M}(P)$

We consider a family of translations that can all be understood as arising from a common pattern that we refer to as $K_{T,M}(P)$ where T and M are two parameters: a set of *tags* and a set of *merges*. A tag $t \in T$ is a set of literals L from P whose truth value is not known in the initial situation I. The tagged literals KL/t in the translation $K_{T,M}$, where L is a literal in P and $t \in T$ is a tag, capture the conditional 'it is known that if t is true initially, then L is true', which we would write in logic as $K(t_0 \supset L)$. The key departure from tagged literals in $K(P)$ is that now tags can be *sets* of literals and they all refer to conditions in the *initial situation* only. Syntactically, we write KL/t rather than L/t because there is a distinction between $\neg KL/t$ and $K\neg L/t$: roughly $\neg KL/t$ means that the conditional $K(t_0 \supset L)$ is not true, while $K\neg L/t$ means that the conditional $K(t_0 \supset \neg L)$ is true.

Each merge $(R, L) \in M$ is a pair where $R \subseteq T$ is a collection of tags in T and L is a literal in P. A merge $m = \langle R, L \rangle$ is *valid* when one of the tags $t \in R$ must be true in I; i.e., when

$$I \models \bigvee_{t \in R} t .$$

We assume that all merges are valid in this sense. Merges (R, L) will map into 'merge actions' $m_{R,L}$ with effects

$$m_{R,L} : \bigwedge_{t \in R} KL/t \to KL .$$

We assume that T always includes a tag t that stands for the empty collection of literals, that we call the *empty tag*. If t is empty, we denote KL/t simply as KL. Similarly, for a set (conjunction) C of literals $L_1, L_2, \ldots, KC/t$ stands for $KL_1/t, KL_2/t, \ldots$, while $\neg K\neg C/t$ stands for $\neg K\neg L_1/t, \neg K\neg L_2/t, \ldots$.

The general translation $K_{T,M}(P)$ is nothing but the core $K_0(P)$ of the $K(P)$ translation 'conditioned' with the tags t in T and extended with the merges in M:

Definition 1 ($K_{T,M}(P)$) *Let $P = \langle F, O, I, G \rangle$ be a conformant problem, then $K_{T,M}(P) = \langle F', I', O', G' \rangle$ is defined as:*[2]

- $F' = \{KL/t, K\neg L/t \mid L \in F \text{ and } t \in T\}$
- $I' = \{KL/t \mid \text{if } I \models t \supset L\}$
- $G' = \{KL \mid L \in G\}$
- $O' = \{a : KC/t \to KL/t, \ a : \neg K\neg C/t \to \neg K\neg L/t \mid a : C \to L \text{ in } P\} \cup \{m_{R,L} : [\bigwedge_{t \in R} KL/t] \to KL \mid (R, L) \in M\}$

where KL is a precondition of action a in $K_{T,M}(P)$ if L is a precondition of a in P.

The translation scheme $K_{T,M}(P)$ reduces to the core translation $K_0(P)$ in $K(P)$ when the set of merges M is empty, and the set of tags T contains only the empty tag. On the other hand, for suitable choices of T and M, we will see that the new translation scheme is *complete*, and under certain conditions, both *complete and polynomial*. At the same time the scheme is simpler than $K(P)$, as there is no need to keep track of flags, nor are there **Split** and **Protection** rules. The lack of flags will also pay off computationally when

---

[2]As in $K_0(P)$, for every rule $a : C/t \to KL/t$ introduced in $K_{T,M}(P)$, the rule $a : C/t \to \neg K\neg L/t$ is added for enforcing the invariant $\neg(KL/t \wedge K\neg L/t)$.

Problem	#S_0	K_{S0}	KP	POND	CFF
Bomb-10-1	1k	648,9	0	1	0
Bomb-10-5	1k	2795,4	0,1	3	0
Bomb-10-10	1k	5568,4	0,1	8	0
Bomb-20-1	1M	$> 1.8G$	0,1	4139	0
Sqr-4-16	4	0,3	fail	1131	13,1
Sqr-4-20	4	0,7	fail	$> 2h$	73,7
Sqr-4-24	4	1,6	fail	$> 2h$	321
Sqr-4-48	4	57,5	fail	$> 2h$	$> 2h$
Safe-50	50	4,3	0,1	9	29,4
Safe-70	70	15	0,1	41	109,9
Safe-100	100	58,7	0,3	5013	1252,4
Sortnet-6	64	2,2	fail	2,1	fail
Sortnet-7	128	27,9	fail	17,98	fail
Sortnet-8	256	$> 1.8G$	fail	907,1	fail
UTS-k08	16	5,1	0,8	429	4,4
UTS-k09	18	9,2	1,4	7203	8,6
UTS-k10	20	15,4	2,1	2426	16,5

Table 1: Illustration of K_{S0} Translation in comparison with KP, POND, and Conformant FF. $K_{S0}(P)$ and $K(P)$ translations fed into FF. For $K(P)$ 'fail' means that FF found the $K(P)$ problem unsolvable; for CFF means goal syntax not handled.

solving $K(P)$ as the heuristics of many classical planners, including FF, ignore deletes. While some instances of the scheme are complete, all of them are sound:

Theorem 2 (Soundness $K_{T,M}(P)$) *If π is a plan that solves the classical planning problem $K_{T,M}(P)$, then the action sequence π' that results from π by dropping the merge actions is a plan that solves the conformant planning problem P.*

Example 2 For the variation of the problem above where the translation $K(P)$ has no plans, let us consider now the translation $K_{T,M}(P)$ with $T = \{at_1, at_2\}$, and the single merge $(T, L) \in M$ with $L = at_3$ that is valid as $at_1 \vee at_2$ is true in I. We can show now that the plan π'_2

$$\{pick(l_1), drop(l_3), pick(l_2), drop(l_3), m_{T,L}\}$$

for $L = at_3$ solves the classical problem $K_{T,M}(P)$ and hence, from Theorem 2, that the plan π_2 obtained from π'_2 by dropping the merge action, is a valid conformant plan for P. We can see how some of the literals in $K_{T,M}(P)$ evolve as the actions in π'_2 are done:

$0: Kat_1/at_1, Kat_2/at_2$	true in I'
$1: Khold/at_1, Kat_2/at_2$	true after $pick(l_1)$
$2: Kat_3/at_1, Kat_2/at_2$	true after $drop(l_3)$
$3: Kat_3/at_1, Khold/at_2$	true after $pick(l_2)$
$4: Kat_3/at_1, Kat_3/at_2$	true after $drop(l_3)$
$5: Kat_3$	true after merge action $m_{T,L}$

where the merge (T, L) is the action with the conditional effect

$$Kat_3/at_1 \wedge Kat_3/at_2 \rightarrow Kat_3$$

whose condition is true before Step 5 producing Kat_3

The Translation $K_{S0}(P)$

A *complete* instance of the translation scheme $K_{T,M}(P)$ can be obtained in a simple manner by setting

- T to the union of the empty tag and the set of possible initial states s_0 (understood as the maximal sets of literals that are consistent with I), and

- M to the merges (T', L) with T' equal to T with the empty tag removed, and L ranging over the literals in P acting as goals or as action preconditions (as both must be achieved with certainty in valid conformant plan)

We will denote this instance, where the tags t range over the possible initial states s_0, as $K_{S0}(P)$. This translation is not only sound but it is also complete:

Theorem 3 (Completeness of $K_{S0}(P)$) *If π is a plan that solves the classical planning problem $K_{S0}(P)$ then the action sequence π' that results from π by dropping the merge actions is a plan that solves the conformant planning problem P, and vice versa, if π is a conformant plan for P, then there is an action sequence π' that extends π with merge actions that is a plan for $K_{S0}(P)$.*

The significance of this result is not only theoretical. There are plenty of conformant problems that are quite hard for current planners even if they involve a handful of possible initial states only. An example of this is the 'Square-Center-n' task (Cimatti, Roveri, & Bertoli 2004), where an agent has to reach the center of an empty square grid with certainty, not knowing its initial location. There are four actions that move the agent one unit in each direction, except when in the border of the grid, where they have no effects. In the standard version of the problem, the initial position is fully unknown resulting in n^2 possible initial states, yet the problem remains difficult, and actually beyond the reach of most planners for small values of n even when the uncertainty is reduced to *a pair possible initial states*. The reason is that then the agent must locate itself before heading for the goal. The same occurs in many other conformant problems.

Table 1 shows results for a conformant planner based on the $K_{S0}(P)$ translation that uses FF (Hoffmann & Nebel 2001) for solving the resulting classical problem, comparing it with three of the four planners that entered the Conformant track of the last competition (Bonet & Givan 2006): POND (Bryce, Kambhampati, & Smith 2006), Conformant FF (Brafman & Hoffmann 2004), and KP (Palacios & Geffner 2006), which uses the $K(P)$ translation also with FF. Clearly, the approach based on the $K_{S0}(P)$ translation does not scale up to problems with many possible initial states, yet the number of such states is small, it does quite well. The $K(P)$ translation yields better results in many domains, but not in the IPC problem 'sortnet' that cannot be solved and where POND does best, and in Sqr-4-n that produces an encoding that FF reports unsolvable (recall that the translations are incomplete). This is a variation of Square-Center-n where the only possible initial locations are the four corners.

Width: Exploiting Relevance and Structure

The translation $K_{S0}(P)$ introduces a number of literals KL/t that is exponential in the worst case: one for each possible initial context t. Yet, planning problems have a structure and it is possible and common that many of the literals L' in a context t are relevant to some literals but irrelevant to others. If for each literal L we could then remove all literals L' that are irrelevant to L from the tags t in the translation $K_{S0}(P)$, we could end up with a polynomially bounded number of tagged literals KL/t while retaining completeness. We will show that this is actually possible in almost all existing benchmarks, but let us first make precise the notion of (conformant) relevance.

Definition 4 (Conformant Relevance) *The relevance relation $L \longrightarrow L'$ in P, read L is relevant to L', is defined inductively as*

1. $L \longrightarrow L$
2. $L \longrightarrow L'$ if $a : C \to L'$ in P with $L \in C$
3. $L \longrightarrow L'$ if $L \longrightarrow L''$ and $L'' \longrightarrow L'$
4. $L \longrightarrow L'$ if $L \longrightarrow \neg L''$ and $L'' \longrightarrow \neg L'$
5. $L \longrightarrow L'$ if both $\neg L$ and L' in a clause in I.

The first clause defining relevance stands for reflexivity, the third for transitivity, the second captures conditions relevant to the effect, and the last captures deductive relevance in the initial situation. The fourth clause, which is the least obvious, captures conditions under which L preempts conditional effects that may delete L'. If we replace 4 by

4' $L \longrightarrow L'$ if $\neg L \to \neg L'$

which is equivalent to 4 in the context of 1–3, and exclude 5, the resulting definition would be precisely the one in (Son & Tu 2006), where the notion of relevance is used to generate a limited set of possible 'partial' initial states over which the 0-approximation is complete. Here we follow a different path, focused on the conditions under which the number of the required tags as opposed to the number of such 'partial' initial states can be polynomially bounded, as the former can be bounded when the latter is not.[3]

Notice that according to the definition a precondition p of an action a is not taken to be 'relevant' to an effect q. The reason is that we want the relation $L \longrightarrow L'$ to capture the conditions under which *uncertainty about L* is relevant *to the uncertainty about L'*. This is why we say this is a relation of *conformant relevance*. Preconditions must be known to be true in order for an action to be applied, so they do not introduce nor propagate uncertainty into the effects of the action.

In order to have polynomial but complete translations we need to make certain assumptions about the formulas in the initial situation I of P. Otherwise, just checking whether

---

[3]In particular, when each literal L unknown in the initial situation is part of a clause that is relevant to some goal or precondition, the number of possible 'partial' initial states to consider in (Son & Tu 2006) is exponential, yet as we will see, this does not imply that the conformant width of the problem, and hence, the number of tags needed, is not bounded. Actually, this is a common situation in the existing benchmarks.

a goal is true in I is intractable by itself and therefore a polynomial but complete translation into classical planning would be impossible (unless P = NP). We will thus assume that I is in *prime implicate (PI) form* (Marquis 2000), meaning that I includes only the inclusion-minimal clauses that it entails but no tautologies. It is known that checking whether a clause follows logically from a formula I in PI form reduces to checking whether the clause is subsumed by a clause in I and hence is polynomial. The initial situations I in most benchmarks is in PI form or can easily be cast into it as they are given by a set of literals and non-overlapping one-of expressions $oneof(X_1, \ldots, X_n)$ that translate into clauses $X_1 \vee \cdots \vee X_n$ and binary clauses $\neg X_i \vee \neg X_j$ for $i \neq j$, so that any resolvent is a tautology.

Provided then with an initial situation I in PI form, let C_I stand for the set of clauses representing uncertainty about I: these are the non-unit clauses in I along with the tautologies $L \vee \neg L$ for complementary literals L and $\neg L$ not appearing as unit clauses in I. We extend the notion of (conformant) relevance to such clauses as follows:

Definition 5 (Relevant Clauses) *A clause $c \in C_I$ is relevant to a literal L in P if all literals $L' \in c$ are relevant to L. The set of clauses in C_I relevant to L is denoted $C_I(L)$.*

Let us also say that a clause $c \in C_I$ *subsumes* another clause $c' \in C_I$, written $c \preceq c'$, if for every literal $L \in c$ and for some literal $L' \in c'$, $I \models L \supset L'$, and let us keep in $C_I^*(L)$, a minimal set of clauses from $C_I(L)$, such that all clauses in $C_I(L)$ are subsumed by a clause in $C_I^*(L)$. The set $C_I^*(L)$ is not necessarily unique but all such sets are *equivalent* and have the same size, which we denote as $|C_I^*(L)|$. Indeed, the relation $c \preceq c'$ partitions $C_I(L)$ into equivalence classes that can be ordered by the relation $\prec$, where $c \prec c'$ if $c \preceq c'$ and $c' \not\preceq c$, so that $|C_I^*(L)|$ is the number of classes in $C_I(L)$ that are minimal with respect to $\prec$.

We can then define the *conformant width* parameter $w(P)$ of a problem P in terms of the number of 'irredundant' clauses in $C_I(L)$ over all preconditions and goal literals L:

Definition 6 (Conformant Width) *Let the* width of a literal *L in P, written as $w(L)$, be $w(L) = |C_I^*(L)|$, and let the* width of the conformant problem *P, $w(P)$, be the max width of any* precondition or goal literal *L*.

We will get then that if the width of a conformant problem P is bounded, i.e., it does not grow with the number of variables in the problem, then it is possible to choose the set of tags T and merges M in the translation $K_{T,M}(P)$ so that the translation becomes *both* polynomial and complete.

This will be an important result as most of the existing benchmarks in conformant planning, with few exceptions, have bounded width, and moreover width equal to 1, meaning that for each precondition or goal literal L there is at most one (unsubsumed) non-unit clause in the initial situation that is relevant to L. More precisely, among the existing benchmarks, the *only ones having conformant width greater than 1* are Blocks, the Adder problem in the last IPC, and Sortnet (in its IPC encoding). All the others, including the ones we consider below in the experiments, have conformant width 1.

Problem	P				$K_1(P)$			
	#Acts	#Atoms	#Effects	Translation time (secs)	#Acts	#Atoms	#Effects	PDDL size
Bomb-100-100	10100	402	40200	1,36	10300	1304	151700	11,77
Sqr-64-ctr	4	130	504	2,34	8	16644	58980	6,1
Sqr-120-ctr	4	242	952	12,32	8	58084	204692	21,7
Logistics-4-10-10	3820	872	7640	1,44	3830	1904	16740	0,9
1-Dispose-8-3	352	486	1984	26,72	361	76236	339410	60,2
Look-n-Grab-8-1-1	352	356	2220	4,03	353	9160	151630	25,3

Table 2: Translation Data: #Effects stands for number of conditional effects $a : C \rightarrow L$. PDDL size is in Megabytes.

The translation scheme $K_i(P)$ that implements such a choice of tags and merges in $K_{T,M}(P)$, uses tags of size i and is complete for problems with width $w(P) \leq i$.

We note that while the results above imply that a problem with low conformant width is relatively 'easy', they do not imply that a problem with a high width is necessarily hard. Indeed, we have found that it is possible to reformulate the Sortnet encoding given in the IPC, into a version with quadratically more fluents that can be solved with the $K_0(P)$ translation. This new version involves the fluents '$i < j$', that represent that content at wire i is less than content at wire j, and 'compare and swap' actions that take i and j as arguments, swapping the contents of the two wires if $i > j$.

The translation $K_0(P)$ in (Palacios & Geffner 2006) is equivalent to the translation $K_i(P)$ for $i = 0$ which is complete for problems with zero $width$. These are precisely the problems where the disjunctive information plays no role:

Theorem 7 (Completeness of $K_0(P)$) *The translation $K_0(P)$ is sound and complete for problems P with width 0.*

The notion of relevance above is actually defined and used in (Son & Tu 2006) where a similar result shows that the 0-approximation semantics is complete for problems P with what we call width 0. This result is equivalent to Theorem 7 as the $K_0(P)$ translation is sound and complete with respect to the 0-approximation semantics (Palacios & Geffner 2006).

The Translation $K_i(P)$

The translations $K_i(P)$, where the parameter i is a nonnegative integer, are complete for problems with width no greater than i and have a complexity that is exponential only in i. In order to define them, let us first recall that C_I stands for the set of non-unit clauses in I along with the tautologies $L \vee \neg L$ for complementary literals L and $\neg L$ that do not appear as unit clauses in I.

We will denote by $C_I^i(L)$ the collection of maximal subsets of $C_I^*(L)$ containing at most i clauses. If the width of P is no greater than i, $C_I^i(L)$ will contain thus a single collection which is $C_I^*(L)$ itself. Also a *cover* t for a set of clauses S stands for any minimal set of literals that contains one literal from every clause in S, and the cover t is consistent if $I \not\models \neg t$. This test is polynomial as we are assuming that I is in PI form. Notice also that a cover t for a set of clauses $S \in C_I^i(P)$ will have always size no greater than i.

Definition 8 *The translation $K_i(P)$ is obtained from the general scheme $K_{T,M}$ by setting*

- T *to the union of the empty tag and the set of consistent covers of any set $S \in C_I^i(L)$ for any precondition or goal literal L, and*
- M *to the set of merges (R, L) for each a precondition and goal literal L and set R given by the collection of all consistent covers of a set $S \in C_I^i(L)$*

For making this definition more transparent, note that this definition implies that $K_i(P)$ for $i = 1$ is $K_{T,M}$ where

- T is the union of the empty tag and the set of literals L (i.e., singletons) in some clause in $C_I(L)$ for some precondition or goal literal L,
- M is the set of merges (R, L) where L is a precondition or goal literal L and R is the set of literals in a clause in $C_I(L)$.

The translation $K_i(P)$ applies to problems P of any width, remaining in all cases exponential only in i but polynomial in both the number of fluents and actions in P. For problems with widths bounded by i, the translation is complete:

Theorem 9 (Completeness of $K_i(P)$) *For conformant problems P with width bounded by i, the translation $K_i(P)$ is sound, complete, and exponential only in i.*

The soundness is the result of the merges M above being valid; the completeness, on the other hand, can be proved by establishing a theoretical equivalence between the $K_i(P)$ translation and the complete but exhaustive translation $K_{S0}(P)$ where the tags t associated with the possible initial states of P are reduced to tags t' of size no greater than i. In problems with conformant width $w(P) \leq i$, it is possible to prove that if a plan in $K_i(P)$ does not achieve KL/t' then the corresponding plan in $K_{S0}(P)$ does not achieve KL/t for some possible initial state $s_0 = t$ with $t' \subseteq t$. Such s_0 can be chosen to satisfy t', the literals implied by t' in the initial situation, and literals that are not relevant to L. For this, the assumption that I is in PI form is needed.

Experimental Results

The conformant planner T_0 is a version of the $K_i(P)$ translation for $i = 1$ optimized for performance, combined with the FF classical planner v2.3 (Hoffmann & Nebel 2001). The $K_1(P)$ translation is provably complete for problems with width 1, but may also solve problems with higher widths as well, although without guarantees. It assumes on the other hand that the initial situation I is in prime-implicate form,

problem	T_0	len	KP	len	CFF	len
Bomb-100-60	5,6	140	4,54	140	9,38	140
Bomb-50-50	1,11	50	0,96	50	0,1	50
Sqr-4-ctr	0,05	8	0,05	8	0,01	12
Sqr-8-ctr	0,07	26	0,05	0	70,63	50
Sqr-12-ctr	0,1	32	0,07	32	$> 2h$	
Sqr-64-ctr	10,68	188	1,66	188	$> 2h$	
Sqr-120-ctr	$> 1.8G$		13,23	356	$> 1.8G$	
Sqr-4-16-ctr	0,2	86	fail		13,13	140
Sqr-4-20-ctr	0,51	128	fail		73,73	214
Sqr-4-24-ctr	1,13	178	fail		320,9	304
Sqr-4-64-ctr	267,3	1118	fail		$> 2h$	
Log-3-10-10	3,42	109	2,67	109	4,67	108
Log-4-10-10	6,52	125	3,07	125	4,36	121
Ring-4	0,09	13	fail		1,37	26
Ring-5	0,1	17	fail		27,35	45
Safe-100	0,18	100	0,26	100	1252,3	100
Safe-50	0,09	50	0,09	50	29,37	50
Safe-70	0,11	70	0,14	70	109,92	70
UTS-K10	1,09	58	2,11	59	16,53	58
UTS-L10	0,33	88	$> 2h$		1,64	59
Comm-21	0,39	313	fail		10,39	269
Comm-22	0,51	348	fail		17,31	299
Comm-23	0,61	383	fail		27,04	329
Comm-24	0,7	418	fail		37,52	359
Comm-25	0,84	453	fail		56,13	389

Table 3: Plan times in seconds and lengths over standard domains. 'fail' means that KP problem reported unsolvable by FF

.

something that T_0 does not check or enforce. The most important optimizations in the $K_1(P)$ translation come from exploiting the notion of relevance further. All the translation schemes above are *uniform* in the sense that the same set of tags T is used over all the literals in P. Yet, whenever a tagged literal KL/t has a tag t that includes literals L' that are not relevant to L, such literals can be removed from t so that literals KL/t get encoded by means of tagged literals KL/t' where t' is the relevant part of t.

The empirical results below are over instances taken from the Conformant-FF distribution, from the recent competition (Bonet & Givan 2006), and from (Palacios & Geffner 2006), with some variations added in the latter case, retaining the encodings in all cases. The experiments were run on a Linux machine running at 2.33 GHz with 8GB of RAM, with a cutoff of 2h or 1.8GB of memory.

Table 2 shows data concerning the translation of a number of instances. The number of conditional effects grows considerably in all cases, and the translation takes several seconds in some cases. E.g., in problem 1-Dispose-8-3, to be explained below, the translation takes almost 28 seconds, yet this is a hard problem that no other planner appears to solve.

Table 3 shows the plan times and lengths obtained by three conformant planners on several standard domains: T_0, $K(P)$ (Palacios & Geffner 2006), and Conformant FF (Brafman & Hoffmann 2004). We have tried also POND (Bryce, Kambhampati, & Smith 2006) but did not perform as well

as Conformant FF. We had more trouble using KACMBP (Cimatti, Roveri, & Bertoli 2004) that does not use the same syntax. In all these domains, T_0 scale up very well with the exception of the Sqr-n-ctr family, where the task is to get to the middle of a grid of size nxn without having any information about the initial location. Here KP does best. Surprisingly, though, when the set of possible initial locations is restricted to the four corners as in the Sqr-4-n-ctr family, KP produces encodings without solutions.

The problems reported in Table 4 are variations of a family of grid problems in (Palacios & Geffner 2006). Dispose is about retrieving objects whose initial location is unknown and placing them in a trash can at a given, known location; Push-to is a variation where objects can be picked up only at two designated positions in the grid to which all objects have to be pushed to: pushing an object from a cell into a contiguous cell moves the object if it is in the cell. 1-Dispose is a variation of Dispose where the robot hand being empty is a condition for the pick up actions to work. As a result, a plan for 1-Dispose has to scan the grid, performing pick ups in every cell, followed by excursions to the trash can, and so on. The plans can get very long (a plan is reported with 1268 actions). Finally, Look-n-Grab is the most interesting problem: the look-n-grab action picks up the objects that are sufficiently close if there are any, and after each pick-up must dump the objects it possibly collected in the trash can before continuing. For the problem P-n-m above, n is grid size and m is the number of objects. For Look-n-Grab, the third parameter is the radius of the action: 1 means that picks up all the objects in the 8 surrounding cells, 2 that picks up all the objects in the 15 surrounding cells, and so on. In practically all these instances, the $K_1(P)$ translation works better than $K(P)$, which in certain cases is just incomplete (where marked 'failed'). We do not include Conformant FF in the table because it only solves 3 instances: Push-to-4-1/2/3. We are not aware of any other planner capable of solving these instances that are rather sophisticated. Actually, among all the instances in Tables 3 and 4, the only problems with *conformant width* different than 1 are 1-Dispose and Look-n-Grab. This is because, the hand being empty is a fluent that is relevant to the goal, and the init clauses about the location of the objects are all relevant to 'hand empty'. Still, while the width is not 1, the $K_1(P)$ translation underlying T_0, makes such problems solvable. For all other problems *the width is 1 because all the actions deal with single objects for which the only relevant clause is the one expressing uncertainty about the object.* This is not the case in problems such as Blocks, where the actions create interactions, but is a common pattern among the available benchmarks.

Summary

While few practical problems are purely conformant, the ability to find conformant plans fast appears to be a necessity in contingent planning where conformant situations are an special case. In this paper, we have extended earlier work, introducing a novel and general translation scheme that maps conformant problems into classical problems that can be both polynomial and complete. The translation scheme depends on two parameters: a set of tags, referring to lo-

cal contexts in the initial situations, and a set of merges that stand for exhaustive sets of tags. We have seen how different translations can be obtained from suitable choices of tags and merges, have introduced a measure of complexity in conformant planning called *conformant width*, and have introduced a translation scheme $K_i(P)$ that involves only tags of size i that is complete for problems of width $\leq i$. We have also shown that almost all conformant benchmarks have width 1, have developed a conformant planner based on the $K_i(P)$ translation that uses the FF classical planner, and have shown that this planner exhibits good performance over the existing domains and some challenging new domains. We are currently working on the use of these ideas for contingent planning.

Acknowledgements

References

Baral, C., and Son, T. C. 1997. Approximate reasoning about actions in presence of sensing and incomplete information. In *Proc. ILPS 1997*, 387–401.

Bonet, B., and Geffner, H. 2000. Planning with incomplete information as heuristic search in belief space. In *Proc. of AIPS-2000*, 52–61. AAAI Press.

Bonet, B., and Givan, B. 2006. Results of the conformant track of the 5th int. planning competition. At http://www.ldc.usb.ve/ bonet/ipc5/docs/results-conformant.pdf.

Brafman, R., and Hoffmann, J. 2004. Conformant planning via heuristic forward search: A new approach. In *Proc. ICAPS-04*.

Bryce, D.; Kambhampati, S.; and Smith, D. E. 2006. Planning graph heuristics for belief space search. *Journal of AI Research* 26:35–99.

Cimatti, A.; Roveri, M.; and Bertoli, P. 2004. Conformant planning via symbolic model checking and heuristic search. *Artificial Intelligence* 159:127–206.

Goldman, R. P., and Boddy, M. S. 1996. Expressive planning and explicit knowledge. In *Proc. AIPS-1996*.

Haslum, P., and Jonsson, P. 1999. Some results on the complexity of planning with incomplete information. In *Proc. ECP-99, Lect. Notes in AI Vol 1809*, 308–318. Springer.

Hoffmann, J., and Brafman, R. 2005. Contingent planning via heuristic forward search with implicit belief states. In *Proc. 15th Int. Conf. on Automated Planning and Scheduling (ICAPS 2005)*, 71–80. AAAI.

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

Marquis, P. 2000. Consequence finding algorithms. In Gabbay, D., and Smets, P., eds., *Handbook on Defeasible Reasoning and Uncertainty Management Systems*, volume 5. Kluwer. 41–145.

Palacios, H., and Geffner, H. 2006. Compiling uncertainty away: Solving conformant planning problems using a classical planner (sometimes). In *Proc. AAAI-06*.

Petrick, R., and Bacchus, F. 2002. A knowledge-based approach to planning with incomplete information and sensing. In *Proc. AIPS'02*, 212–221.

Rintanen, J. 2004. Complexity of planning with partial observability. In *Proc. ICAPS-2004*, 345–354.

Smith, D., and Weld, D. 1998. Conformant graphplan. In *Proceedings AAAI-98*, 889–896. AAAI Press.

Son, T. C., and Tu, P. H. 2006. On the completeness of approximation based reasoning and planning in action theories with incomplete information. In *Proc. 10th Int. Conf. on Principles of KR and Reasoning (Kr-06)*, 481–491.

Son, T. C.; Tu, P. H.; Gelfond, M.; and Morales, A. 2005. Conformant planning for domains with constraints: A new approach. In *Proc. AAAI-05*, 1211–1216.

Problem	T0	len	Kp	len
Dispose-12-1	17,77	709	14,02	683
Dispose-12-2	39,62	811	317,42	826
Dispose-12-3	> 1.8G		2434,04	985
Dispose-16-1	60,34	1357	454,91	1361
Dispose-16-2	818,2	1748	4764,13	1680
Dispose-16-3	> 1.8G		> 1.8G	
Dispose-20-1	174,71	1926	> 1.8G	
Dispose-20-2	> 1.8G		> 1.8G	
Push-to-4-1	0,16	64	> 1.8G	
Push-to-4-2	0,3	67	0,16	69
Push-to-4-3	0,48	83	0,22	71
Push-to-8-1	63,23	369	> 1.8G	
Push-to-8-2	928,63	452	5,7	289
Push-to-8-3	1153,16	395	10,12	291
Push-to-12-1	> 2h		> 1.8G	
1-Dispose-4-1	0,21	140	fail	
1-Dispose-4-2	0,68	140	fail	
1-Dispose-4-3	1,82	140	fail	
1-Dispose-8-1	124,5	1268	fail	
1-Dispose-8-2	699,11	1268	fail	
1-Dispose-8-3	1296,02	1268	fail	
1-Dispose-12-1	> 2h		fail	
Look-n-Grab-4-1-1	0,31	26	fail	
Look-n-Grab-4-2-1	1,49	26	fail	
Look-n-Grab-4-3-1	5,12	26	fail	
Look-n-Grab-8-1-1	45,27	212	fail	
Look-n-Grab-8-1-2	84,04	88	fail	
Look-n-Grab-8-2-1	> 1.8G		fail	

Table 4: Challenging Problems over a Grid: plan times in seconds and lengths shown. 'fail' means that FF found $K(P)$ problem unsolvable. Figures for Conformant-FF not included, as it solves only 3 instances: Push-to-4-1/3.