# Translation based approaches to probabilistic planning

**Ran Taig**
Department of Computer Science
Ben Gurion University of The Negev
Beer-Sheva, Israel 84105
taig@cs.bgu.ac.il

## Abstract

The main focus of our work is the use of classical planning algorithms in service of more complex problems of planning under uncertainty. In particular, we are exploring compilation techniques that allow us to reduce some probabilistic planning problems into variants of classical planning, such as metric planning, resource-bounded planning, and cost-bounded suboptimal planning. Currently, our initial work focuses on *conformant probabilistic planning*. We intend to improve our current methods by improving our compilation methods, but also by improving the ability of current planners to handle the special features of our compiled problems. Then, we hope to extend these techniques to handle more complex probabilistic settings, such as problems with stochastic actions and partial observability.

## Motivation

Models of planning under uncertainty, and in particular, MDPs and POMDPs have received much attention in the AI and Decision-Theoretic planning communities (Boutilier, Dean, and Hanks 1999; Kaelbling, Littman, and Cassandra 1998). These models allow for a richer and more realistic representation of real-world planning problems, but lead to increased complexity. Recently, a new approach for handling certain simple classes of planning under uncertainty was introduced (Palacios and Geffner 2009). This approach works by reducing problems of planning under uncertainty to classical planning problems. The main benefit of this technique is that it allows us to exploit techniques developed in classical planning, and in particular, effective and sophisticated methods for computing heuristic functions. So far, this technique has been shown to be effective for conformant and contingent planning (Albore, Palacios, and Geffner 2009; Shani and Brafman 2011). A related approach was very successful in handling MDPs in the FF-Replan planner (Yoon, Fern, and Givan 2007). But whereas the compilation method attempt to capture information about the state of knowledge of the agent within the classical planning problem, the methods used by FF-Replan are based on simple determinzation of the problem, and the classical planning problem ignores the uncertainty inherent to the problem. In addition, the model used assumes perfect observability, and so following each action, the agent has complete information about the current state of the world.



The goal of our research is to investigate extensions of the ideas of problem compilation to probabilistic planning problems. That is, we'd like to be able to represent within the classical planning problem information about the probability of success. To start this research direction, we focus on the problem of conformant probabilistic planning with deterministic actions. Although this problem is not too exciting, much like conformant planning, it provides a convenient initial step for exploring this research direction. We believe that our techniques can be extended to more general probabilistic planning problems.

## Conformant Probabilistic Planning (CPP)

Conformant probabilistic planning tasks are quadruples $(A, b_I, G, \theta)$, corresponding to the *action set*, *initial belief state*, *goals*, and *acceptable goal satisfaction probability*. As before, $G$ is a set of propositions. The initial state is no longer assumed to be known precisely. Instead, we are given a probability distribution over the world states, $b_I$, where $b_I(w)$ describes the likelihood of $w$ being the initial world state. In its most general form, CPP covers stochastic actions as well, but we leave this to future work.

There is no change in the definition of actions and their applications in states of the world comapring to the usual classical planning formulations. But since we now work with belief states, actions can also be viewed as transforming one belief state to another. The likelihood $[b, a](w')$ of a world state $w'$ in the belief state $[b, a]$, resulting from applying action $a$ in belief state $b$, is given by

$$[b, a](w') = \sum_{a(w)=w'} b(w) \tag{0.1}$$

We will also use the notation $[b, a](\varphi)$ to denote $\sum_{a(w)=w', w' \models \varphi} b(w)$, and we somewhat abuse notation and write $[b, a] \models \varphi$ for the case where $[b, a](\varphi) = 1$.

For any action sequence $\overline{a} \in A^*$, and any belief state $b$, the new belief state $[b, \overline{a}]$ resulting from applying $\overline{a}$ at $b$ is

given by

$$[b, \overline{a}] = \begin{cases} b, & \overline{a} = \langle\rangle \\ [b, a], & \overline{a} = \langle a \rangle, a \in A \\ [[b, a], \overline{a}'], & \overline{a} = \langle a \rangle \cdot \overline{a}', a \in A, \overline{a}' \neq \emptyset \end{cases} . \quad (0.2)$$

In such setting, achieving $G$ with certainty is typically unrealistic. Hence, $\theta$ specifies the required *lower bound* on the probability of achieving $G$. A sequence of actions $\overline{a}$ is called a *plan* if we have $b_{\overline{a}}(G) \geq \theta$ for the belief state $b_{\overline{a}} = [b_I, \overline{a}]$. Because our actions are deterministic, this is essentially saying that $\overline{a}$ is a *plan* if $Pr(\{w : \overline{a}(w) \models G\}) \geq \theta$, i.e,. the weight of the initial states from which the plan reaches the goal is at least $\theta$.

## Approaches for planning under uncertainty

There are two main approaches for planning under uncertainty, originally introduced in the context of non-deterministic models of uncertainty, commonly used in recent years: planning via search in belief space, and the translation-based method for planning under uncertainty.

**Planning via search in belief space** Uncertainty in planning models show up as uncertainty about the initial state of the world and/or uncertainty about the effects of actions, either non-deterministic or stochastic. In both cases the agent has some "beliefs" during the planning process, about which states it can possibly be in. Formally, a belief state $b$ is the non empty set of states that are deemed possible in a given situation. An action is executable in a belief state $b$ if it is executable (i.e., its preconditions are satisfied) in every state in $b$. Every action $a$ executable in $b$ maps $b$ into a new belief state $b_a = \{s' \mid \text{such that } s' = f(a, s) \text{ for some } s \in b\}$ where $f(a, s)$ denotes the result of applying the action $a$ in a state $s$ according to the transition function.

We denote by $b_I$ the *initial belief state* which consists of all possible plain initial states. A *goal belief state* is any belief state which contains goal states only. These definitions define a search problem in which the initial state is $b_I$ and the operators are the actions. The resulting plan maps all states in $b_I$ into goal state, i.e., it is a conformant plan.

The approach suffers from two main disadvantages. First, on interesting enough planning problems the size of the belief state space is huge and cannot be explicitly represented – it is exponentially larger than the state space. In fact, each belief state can contain an exponential number of states. This problem was dealt using logical representation techniques (SAT, OBDD) which can compactly represent beliefs states, sometimes. These methods were partially successful and formed the basis of most conformant planners, until recently.

A more critical problem is that while heuristic search turned out to be highly successful in classical planning, it was difficult to extend this success to search in belief space which has structure that is harder to approximate via various relaxation methods. The few heuristics developed for the belief space search were much less informative than typical heuristics applied to (standard) state space.

**The translation approach - background and required modifications** We present here a modified version of the translation-based method of (Palacios and Geffner 2009), adapted to our settings. The essential idea behind the translation approach to conformant planning implemented in the $T_0$ planner is to reason by cases. The different cases correspond to different conditions on the initial state, or, equivalently, different sets of initial states. These sets of states, or conditions, are captured by tags. That is, a tag is identified with a subset of $b_I$. Below we abuse notation often, treating a tag as the set of initial states it defines.

With every proposition $p$, we associate a set of tags $T_p$. We require that this set be *deterministic* and *complete*. We say that $T_p$ is *deterministic* if for every $t \in T_p$ and any sequence of actions $\bar{a}$, the value of $p$ is uniquely determined by $t$, the initial belief state $b_I$ and $\bar{a}$. We say that $T_p$ is *complete* w.r.t. an initial belief state $b_I$ if $b_I \subseteq \bigcup_{t \in T_p} t$. That is, it covers all possible relevant cases. We say that a set of tags is *disjoint* when for every $t \neq t' \in T_p$ we have that $t \cap t' = \emptyset$. We say that a set of tags is *DCD* if it is deterministic, complete, and disjoint.

Once we compute the tags required for a proposition $p$, (see below) we augment the set of propositions with new propositions of the form $p/t$, where $t$ is one of the possible tags for $p$. $p/t$ holds the current value of $p$ given that the initial state satisfies the condition $t$. The value of each proposition $p/t$ is known initially – it reflects the value of $p$ in the initial states represented by $t$, and since we focus on deterministic tags only, then $p/t \vee \neg p/t$ is a tautology throughout. Our notation $p/t$ differs a bit from the $Kp/t$ notation of Palacios and Geffner. The latter is used to stress the fact that these propositions are actually representing knowledge about the belief state. However, because of our assumption that tags are deterministic, we have that $\neg Kp \rightarrow K\neg p$. To stress this and remove the redundancy, we use a single proposition $p/t$ instead of two propositions $Kp/t, K\neg p/t$.

The actions are transformed accordingly to maintain our state of knowledge. Given the manner tags were selected, we always know how an action would alter the value of some proposition given any of its tags. Thus, we augment the description of actions to reflect this. If the actions are deterministic (which we assume in this paper), then the change to our state of knowledge is also deterministic, and we can reflect it by altering the action description appropriately.

The resulting problem is a classical planning problem defined on a larger set of variables. The size of this set depends on the original set of variables and the number of tags we need to add. Hence, an efficient tag generation process is important. A trivial set of tags is one that contains one tag for each possible initial state. Clearly, if we know the initial state of the world, then we know the value of all variables following the execution of any set of actions. However, we can often do much better, as the value of each proposition at the current state depends only on a small number of propositions in the initial state. This allows us to use many fewer tags (=cases). In fact, the current value of different propositions depends on different aspects of the initial state. Thus, in practice, we select different tags for each proposition. We generate the tags for $p$ by finding which literals are relevant

to its value using the following recursive definition:
1.$p$ is relevant to $p$.
2. If $q$ appears (possibly negated) in an effect condition $c$ for action $A$ such that $c \rightarrow r$ and $r$ contains $p$ or $\neg p$ then $q$ is relevant to $p$.
3.If $r$ is relevant to $q$ and $q$ is relevant to $p$ then $r$ is relevant to $p$.

Let $C_p$ denote the set containing all the propositions relevant to $p$. The set of tags consisting of one tag for every possible assignment to $C_p$ is DCD. This set can be reduced farther, while remaining DCC, if we remove any tag that corresponds to an assignment to $C_p$ which has probability 0 in the initial state.

## Current work

Our work so far offers new compilations schemes that utilize efficient variants of classical planners to solve conformant probabilistic planning problems (CPP).

We solve these problems by compiling them to variants of classical planning, which can then be solved by existing solvers. Our first method transforms CPP into metric planning, where numeric variables represent information about the belief state of the agent, this work appears as (Brafman and Taig 2011). The second method translates CPP into classical planning with resource constraints, in which the resources represent probabilities of failure. A very similar method translates CPP into cost-optimal classical planning problems. both described in(Taig and Brafman 2012). Finally, our most recent and empirically successful method, to be presented in our ICAPS'13 paper, reduces CPP into cost-bounded suboptimal classical planning problems. Empirically, all techniques show mixed results, performing well on some domains and poorly on others. The bottleneck seems to be the less developed state-of-the-art in metric and resource bounded planning, areas to which many of the recent techniques introduced in classical planning are yet to be integrated. Nevertheless, we believe that these compilation techniques offer an interesting and promising direction for future research on probabilistic planning in structured domains. Due to lack of space we will describe here only the first and last methods.

## Related work

The best current probabilistic conformant planner is *Probabilistic FF* (PFF) (Domshlak and Hoffmann 2007). The basic ideas underlying Probabilistic-FF are:

1. Define time-stamped Bayesian Networks (BN) describing probabilistic belief states.
2. Extend Conformant-FF's belief state to model these BN.
3. In addition to the SAT reasoning used by Conformant-FF (Hoffmann and Brafman 2006), use weighted model-counting to determine whether the probability of the (unknown) goals in a belief state is high enough.
4. Introduce approximate probabilistic reasoning into Conformant-FF's heuristic function.

In many domains, *PFF*'s results were improved by our results. An earlier attempt to deal with probabilities by reducing it to action costs appears in (Jiménez et al. 2006) in the context of probabilistic planning problems where actions have probabilistic effects but there is no uncertainty about the initial state. The probabilistic problem is compiled into a (non-equivalent) classical problem where each possible effect $e$ is represented by a unique action and the cost associated with this action is set to be $1 - Pr(e)$. That value captures the amount of risk the planner takes when choosing that action, which equals the probability that the effect won't take place when the original action is executed. This value is then minimized by the cost-optimal planner. Our compilation scheme uses related ideas but deals with uncertainty about the initial state, and comes with correctness guarantees.

Closely related to our work is the *CLG+* planner (Albore and Geffner 2009). This planner attempts to solve contingent planning problems in which goal achievement cannot be guaranteed. Thus, gradually, this planner makes assumptions that reduce the uncertainty, and allow it to plan. This is achieved by adding special actions, much like ours, that "drop" a tag, i.e., assume its value is impossible. These actions are associated with a high cost. The main difference, of course, with our planner is that the cost we associate with assumption-making actions reflects the probability of the states ruled out, allowing us to model probabilistic planning problems as cost-optimal planning. Furthermore, our planner may decide (depending on the search procedure used) to come up with a sub-optimal plan, albeit one that meets the desired probabilistic threshold, even when a full conformant plan exists. This flexibility allows us to trade-off computational efficiency with probability of success.

Of similar flavor to the above is the assumption-based planning approach introduced recently (Davis-Mendelow, Baier, and McIlraith 2012). This work considers the problem of solving conformant and contingent planning under various assumptions, that may be selected according to various preference criteria. It too does not consider an explicit probabilistic semantics that addresses CPP.

## Method 1: Compiling CPP into Metric Planning

We create a metric planning problem which is then given to *Metric-FF*. The solution plan is returned as a plan for the CPP given as input.

### The Metric Planning Problem

Let $P = (V, A, b_I, G, \theta)$ be the CPP given as input. Recall that $T_p$ is the set of tags for $p$. We use $T$ to denote the entire set of tags (i.e., $\cup T_p$). We generate a metric-planning problem $\widehat{P} = (\widehat{V}, \widehat{F}, \hat{A}, \widehat{I}, \widehat{G})$ as follows:

**Propositions:** $\widehat{V} = \{p/t \mid p \in V, t \in T_p\}$.

**Functions:** $\widehat{F} = \{Pr_p \mid p \in V\} \cup \{Pr_{goal}\}$. That is functions that keep the current probability of each original proposition. We sometimes abuse notation and write $Pr_{\neg p}$ instead of $1 - Pr_p$. Finally $Pr_{goal}$ denotes the probability that the goal is true.

**Numerical Constants:** We use a group $\widehat{c}$ of constants to save the initial probability of each tag $t \in T$: $\widehat{c} = \{b_I(t) \mid t \in T\}$. Note that these can be computed from the initial state description.

**Initial State:**

- $\widehat{I} = \{l/t \mid l$ is a literal, and $t, I \models l, \}$
- $Pr_p = b_I(\{s|s \models p\})$, i.e., the initial probability that $p$ holds.
- $Pr_{goal} = b_I(\{s|s \models G\})$. Again, this can be computed directly from the initial state description.

**Goal:** $\widehat{G} = \{Pr_{goal} \geq \theta\}$.

**Actions:** First, for every action $a \in \hat{A}$, we make all its effects conditionals. Thus, if $e$ is an effect of $a$, we now treat it as a conditional effect of the form $\emptyset \rightarrow \{e\}$.

For every action $a \in A$, $\hat{A}$ contains an action $\widehat{a}$ defined as follows:

- $pre(\widehat{a}) = \{P_l = 1 \mid l \in pre(a)\}$. This reflects the need to make sure actions in the plan are always applicable: The probability of the preconditions is 1 only if they hold given all possible initial states.[1]
- For every conditional effect $(con \rightarrow eff) \in E(a)$, $\widehat{a}$ contains the following conditional effects for each $e \in eff$ and for every $t \in T$:
  - $\{c/t \mid c \in con \cup \{\neg e\}\} \rightarrow \{e/t, Pr_e = Pr_e + b_I(t)\}$. That is, if we know all conditions of the conditional effects are true before applying the action given $t$ is true initially then we can conclude that the effect takes place so we now know that $e$ is true under the same assumption. This information is captured by adding $e/t$. Note that we care only about conditional effects that actually change the state of the world. Hence, we require that the effect not hold prior to the execution of the action. In that case, the new probability of $e$ is the old probability of $e$ plus the probability of the case (as captured by the tag $t$) we are considering now.
  - If $e \in G$ we also add the following effect to the last condition:
    $Pr_{goal} = Pr_{goal} + (b_I(t) \times \prod_{e' \in G \setminus \{e\}} Pr_{e'})$
    If $\neg e \in G$ we add the following effect to the last condition:
    $Pr_{goal} = Pr_{goal} - (b_I(t) \times \prod_{e' \in G \setminus \{e\}} Pr_{e'})$
    If $e \in G$ then our knowledge of the probability of the goal was changed by the action so that now: $Pr^{new}_{goal} = \prod_{e \in G} Pr_e$. Note that here we assume that the probability of the different sub-goals is independent.[2] Given the increase in the probability of $e$ and the independence assumption, the the new goal probability is : $\prod_{e' \in G \setminus \{e\}} Pr_{e'} \times (Pr_e + b_I(t)) = Pr^{old}_{goal} + (b_I(t) \times \prod_{e' \in G \setminus \{e\}} Pr_{e'})$ . The same rational guides us when the action reduces the probability of some sub-goal.

[1] In this scheme we follow the convention of earlier planners requiring that a plan be executable in all initial states.

[2] We can handle the case of dependent goals, but that requires adding more tags, i.e., by adding tags that determinize the goal.

## Accuracy of probabilistic calculations

The soundness of our algorithm rests on the accuracy of our probabilistic estimate of the value of $P_{goal}$. We now prove that this value is correct under the assumption that the set of tags is *deterministic, complete*, and *disjoint*. We defined the notion of deterministic and complete tags earlier. We say that a set of tags $T_p$ is **disjoint** if $\forall t_i, t_j \in T_p$ s.t. $i \neq j$ and for every possible initial state $s_I$: $t_i \models s_I \Rightarrow t_j \not\models s_I$.

**Lemma 1** *Let $\bar{a}$ be a sequence of actions from A, let $\hat{a}$ be the corresponding sequence of actions from $\hat{A}$, and let $t \in T_p$ be a deterministic tag. Then, $[b_I, \bar{a}] \models p/t$ iff for every initially possible world state $w \in t$ we have that $(\hat{a})(w) \models p$.*

This lemma follows from our construction of the new actions, together with the fact that the tags are deterministic, i.e., the value of $p$ in $(\hat{a})(w)$ for all initially possible world states $w \in t$ is the same.

**Lemma 2** *Let $p \in V$, and assume that $T_p$ is* deterministic, complete, *and* disjoint. *Let $\bar{a}$ be a sequence of actions in A. Let $\hat{a}$ be the corresponding sequence in $\hat{A}$. Then, $\hat{a}(Pr_p) = [b_I, \bar{a}]\,(p)$. That is, at this stage, $Pr_p$ equals the probability of p following the execution of $\bar{a}$.*

Due to lack of space,we omit the proof here and refer you to the paper.

**Corollary 3** *The plan returned is a legal plan for P.*

**Proof:** Each action precondition $L$ is replaced by the precondition $P_L = 1$, from lemma 1 we learn that this property holds if and only if $L$ is known with full certainty and the action can be applied. ∎

**Corollary 4** *Assuming that sub-goals are probabilistically independent, then, in each stage of the planning process $Pr_{goal}$ holds the accurate probability of the goal state.*

**Proof:** If $G = \{L\}$ then it's immediate from lemma 2. Otherwise, from lemma 2 it follows that this holds true for every sub-goal. Thus, the probability of the goal is the product of the probability of the sub-goals. The proof follows by induction from the fact that we initialize $Pr_{goal}$ correctly, and from the updates performed following each action. Specifically, suppose that the probability of subgoal $g$ increased following the last action. The new goal probability is : $\prod_{g' \in G \setminus \{g\}} Pr_{g'} \times (Pr_g + b_I(t)) = Pr^{old}_{goal} + (b_I(t) \times \prod_{g' \in G \setminus \{g\}} Pr_{g'})$. By construction, one effect of the corresponding action in $\hat{A}$ is $Pr_{goal} = Pr_{goal} + (b_I(t) \times \prod_{g' \in G \setminus \{g\}} Pr_{e'})$. This maintains the correct value. A similar update occurs in the case of a reduction. Since updates are done sequentially, the value remains correct even if an action affects multiple goals. ∎

## Method 2: CPP as cost-bounded sub optimal planning problem

**Cost bounded classical planning** In cost bounded classical planning a classical planning problem is extended with a constant parameter $c \in \mathbb{R} > 0$. The task is to find a plan with cost $\leq c$ as fast as possible. In this setting the optimal plan cost and the distance of the resulting plan from optimal does not matter, as opposed to notions such as suboptimal search. One way to solve this problem is to use an optimal planner and then confirm that the cost bound is met. Another method is to use an anytime planner that gradually improves the plan cost, until the cost bound is met. However, these methods do not make real use of the bound during the search process, e.g., for pruning nodes that cannot lead to a legal solution. Recently, a number of algorithms that deal directly with this problem were suggested (Stern, Puzis, and Felner 2011),(Thayer et al. 2012). The common ground of all these algorithms is the consideration of the bound $c$ within the heuristic function. One example is the *Potential Search* algorithm that uses heuristic estimates to calculate the probability that a solution of cost no more than $c$ exists below a given node (Stern, Puzis, and Felner 2011). This idea was extended by the *Beeps* algorithm (Thayer et al. 2012) which chooses which node to expand next based on the node's potential which combines admissible and inadmissible estimates of the node's $h$ value as well as an inadmissible estimate of the number of actions left to the goal (distance estimate). This algorithm is currently considered the state of the art for this problem.

**Proposed compilation** The basic motivation for the method we present now is the understanding that we can solve a CPP problem by identifying a set $b'$ of initial states whose joint probability is greater or equal to $\theta$, such that there exists a conformant plan for $b'$. This plan is a solution to the CPP problem, too.

The task of finding a plan to a CPP can thus be divided into the identification of a suitable initial belief state $b_I$, followed by finding a plan to the conformant planning problem $(V, A, b_I, G)$. However, rather than take this approach directly, we use a compilation-based method in which we let the classical planner handle both parts. That is, in the classical planning problem we generate the planner decides which states to ignore, and also generates a conformant plan for all other states. We must ensure that the joint probability of ignored states does not exceed $1 - \theta$. Technically, this is done by introducing special actions that essentially tell the planner to ignore a state (or set of states). The cost of each such action is equal to the probability of the state(s) it allows us to ignore.

Technically, the effect of ignoring a state is to make it easier for the planner to obtain knowledge. Typically, we say that the agent knows $\varphi$ at a certain belief state, if $\varphi$ holds in all world states in this belief state. In the compilation approach such knowledge is added by applying *merge* actions. Once a state has been "ignored" by an "ignore" action, the merge actions effectively ignore it, and deduce the information as if this state is not possible.

**Formal description** Let $P = (V, A, b_I, G, \theta)$ be the input CPP. Let $T_p$ be the set of tags for $p$. (We discuss the tag computation algorithm later – right now it is assumed to be given to the compiler.) We use $T$ to denote the entire set of tags (i.e., $\cup T_p$). We will also assume a special distinguished tag, the empty set. We now present our compilation methods for $P$.

Given $P$, we generate the following classical planning with action costs problem $\widetilde{P} = (\widetilde{V}, \widetilde{A}, \widetilde{I}, \widetilde{G})$:

**Variables:** $\widetilde{V} = \{p/t \mid p \in V, t \in T_p\} \cup \{Drop_t | t \in T\}$. The first set of variables are as explained above. $p/\{\}$, which we abbreviate as simply $p$, denotes the fact that $p$ holds unconditionally, i.e., in all possible worlds. (Previous work denotes this by $Kp$.) The second set of variables – $Drop_t$ – denotes the fact that we can ignore tag $t$.

**Initial State:** $\widetilde{I} = \{l/t \mid l$ is a literal, and $t, I \vDash l\}$. All valid assumptions on the initial worlds captured by the special variables. Note that all $Drop_t$ propositions are false.

**Goal State :** $\widetilde{G} = G$. The goal must hold for all initial states. Recall that what we call knowledge is not real knowledge, because we allow ourselves to overlook the ignored states.

**Actions:** $\widetilde{A} = \widetilde{A}_1 \cup \widetilde{A}_2 \cup \widetilde{A}_3 \cup \widetilde{A}_4$ where:

- $\widetilde{A}_1 = \{\widetilde{a} \mid a \in A\}$: Essentially, the original set of actions.
  - $pre(\widetilde{a}) = pre(a)$. That is, to apply an action, its preconditions must be known.
  - For every conditional effect $(c \rightarrow p) \in E(a)$ and for every $t \in T$, $\widetilde{a}$ contains: $\{c/t \mid c \in con\} \rightarrow \{p/t\}$. That is, for every possible tag $t$, if the condition holds given $t$, so does the effect.
  - $cost(\widetilde{a}) = 0$.
- $\widetilde{A}_2 = \{\{p/t \mid t \in T_p\} \rightarrow p$ |p is a precondition of some action $a \in A$ or $p \in G\}$. These are known as the *merge* actions. They allow us to infer from conditional knowledge about $p$, given certain sets of tags, absolute knowledge about $p$. That is, if $p$ holds given $t$, for an appropriate set of tags, then $p$ must hold everywhere, we set the cost of all the merge actions to 0 as well.
- $\widetilde{A}_3 = \{Drop_t \mid t \in T\}$ where: $pre(Drop_t) = \{\}, eff(Drop_t) = \{Drop_t\}, cost(Drop_t) = Pr_I(t)$. That is, the $Drop_t$ action let's us drop tag $t$, making $Drop_t$ true in the cost of $t$'s initial probability.
- $\widetilde{A}_4 = \{Assume_{p/t}$ |p is a precondition of some action $a \in A$ or $p \in G, t \in T_p\}$ $pre(Assume_{p/t}) = \{Drop_t\}, eff(Assume_{p/t}) = \{p/t\}, cost(Assume_{p/t}) = 0$. That is, we can assume whatever we want about what is true given a dropped tag.

**Cost bound :** The cost bound for the classical planning problem is set to $1 - \theta$.

Thus, essentially, using the $Drop_t$ action, the planner decides to "pay" some probability for dropping all initial states that correspond to this tag. Once it drops a tag, it can conclude whatever it wants given this tag. Thus, achieving the goal given a dropped tag is trivial. A solution to $\widetilde{P}$ will make assumptions whose cost does not exceed the bound. Hence,

it will work on a (probabilistically) sufficiently large set of initial states.

We refer the reader to our ICAPS'13 paper for a detailed example,empirical and theoretical results for this scheme.

## Research Plan

Initially, we intend to focus on the following main research thrusts:

1. Developing and implementing more suitable compilation techniques. Currently, our implementation is based on algorithms developed for conformant planning. We will develop variants of these algorithms that are more suitable for probabilistic planning, for example, by developing appropriate tag generation techniques, a central element of current compilation techniques aimed at reducing the complexity of the classical planning problems generated. Specifically, nowadays we investigate the possibility to apply the the state of the art approach for conformant planning (Nguyen et al. 2012) into probabilistic planning. This method is based on fixing a plan suitable to one possible initial state into all (or enough in our case) states. The probabilistic setting presents a few challenges to the original algorithm such as identifying possible initial states with no possible plan, best ordering of initial states w.r.t their initial probabilities.

2. Developing approximation techniques that handle problems with conformant width greater than 1 - Since we are not able to keep our probabilistic calculations accurate we have to calculate lower bounds on the current probability of the goal. This will keep our schemes sound, but may lead to poor results when the lower bound we calculate is strictly smaller than $\theta$ and the real goal probability is strictly higher. To be more specific, this (conservative) approach avoids adding the probabilistic weight of initial worlds represented by a tag $t$ to a relevant predicate $L$ if $L/t$ is not known for certain, even if $L$ is actually true given $t$. The current tag generation system is based on two possible compilations. The first treats any given domain as having a conformant width of 1 ($K_1$ translation) (Palacios and Geffner 2009). The second is a full translation, representing all possible initial states. The latter is mostly not usable, since both the time and size of translation are exponentially large.

   Our goal is to develop a flexible system, able to calculate the translation $K_i$ given a user input for the parameter $i$. This translation means that we want the tags to consider at most $i$ of the relevant *uncertainty clauses* for each predicate $L$. This kind of flexibility will allow to control the trade off between the complexity of the compilation and the completeness of the algorithm.

   We intend to back-up this system with theoretical guarantees, formalizing the notion of "distance" between the approximation and the real probability. This will enable a user to sacrifice complexity in order to become complete, or on the other hand to sacrifice the soundness in order to achieve fast results, as we can easily calculate upper bounds on the probabilities.

3. Improving the ability of variants of classical planners to handle the resulting problems. Currently, we use these algorithms as black boxes, but an analysis of our empirical results shows the delete-relaxation based heuristic functions they use to handle resources are inadequate for our problems. These heuristics ignore adverse effects of actions, and in particular, reductions in resource values in the case of numeric variable in metric planning. This makes them essentially uninformative. Thus, we will seek new heuristics for classical planning that better handle the class of problems we generate.

4. Extending our compilation techniques to handle richer settings. As a first step, we will focus on CPP with stochastic actions. Later, we hope to handle stochastic observations, motivated by the work done at (Albore, Palacios, and Geffner 2009; Shani and Brafman 2011) where the translation based approach was used to handle *contingent planning* with non-deterministic observations.

## References


Albore, A., and Geffner, H. 2009. Acting in partially observable environments when achievement of the goal cannot be guaranteed. In *ICAPS'09 Planning and Plan Execution for Real-World Systems Workshop*.

Albore, A.; Palacios, H.; and Geffner, H. 2009. A translation-based approach to contingent planning. In *IJCAI*, 1623–1628.

Boutilier, C.; Dean, T.; and Hanks, S. 1999. Decision-theoretic planning: Structural assumptions and computational leverage. *J. Artif. Intell. Res. (JAIR)* 11:1–94.

Brafman, R. I., and Taig, R. 2011. A translation based approach to probabilistic conformant planning. In *ADT*.

Davis-Mendelow, S.; Baier, J.; and McIlraith, S. 2012. Making reasonable assumptions to plan with incomplete information. *HSDIP 2012* 69.

Domshlak, C., and Hoffmann, J. 2007. Probabilistic planning via heuristic forward search and weighted model counting. *J. Artif. Intell. Res. (JAIR)* 30:565–620.

Hoffmann, J., and Brafman, R. I. 2006. Conformant planning via heuristic forward search: A new approach. *Artif. Intell.* 170(6-7):507–541.

Jiménez, S.; Coles, A.; Smith, A.; and Madrid, I. 2006. Planning in probabilistic domains using a deterministic numeric planner. In *The 25th PlanSig WS*.

Kaelbling, L. P.; Littman, M. L.; and Cassandra, A. R. 1998. Planning and acting in partially observable stochastic domains. *Artif. Intell.* 101(1-2):99–134.

Nguyen, H.-K.; Tran, D.-V.; Son, T. C.; and Pontelli, E. 2012. On computing conformant plans using classical planners: A generate-and-complete approach. In *ICAPS*.

Palacios, H., and Geffner, H. 2009. Compiling uncertainty away in conformant planning problems with bounded width. *JAIR* 35:623–675.

Shani, G., and Brafman, R. I. 2011. Replanning in domains with partial information and sensing actions. In *IJCAI*, 2021–2026.

Stern, R.; Puzis, R.; and Felner, A. 2011. Potential search: a bounded-cost search algorithm. In *Proceedings of the Twenty-First International Conference on Automated Planning and Scheduling*.

Taig, R., and Brafman, R. I. 2012. Using classical planners to solve conformant probabilistic planning problems. In *Problem Solving Using Classical Planners AAAI Technical Report WS-12-12,p. 65-71*.

Thayer, J.; Stern, R.; Felner, A.; and Ruml, W. 2012. Faster bounded-cost search using inadmissible estimates. In *Proceedings of the Twenty-Second International Conference on Automated Planning and Scheduling*.

Yoon, S.; Fern, A.; and Givan, R. 2007. Ff-replan: A baseline for probabilistic planning. In *ICAPS*, volume 7, 352–359.