Modeling action feasibility in POMDPs with boolean-valued preconditions

Caroline P. Carvalho Chanel [1,2], **Florent Teichteil-Königsbuch** [2], **Guillaume Infantes** [2] and **Patrick Fabiani** [2]
[1] Université de Toulouse - ISAE - Institut Supérieur de l'Aéronautique et de l'Espace
[2]Onera - The French Aerospace Lab, F-31055, Toulouse, France
name.surname@onera.fr

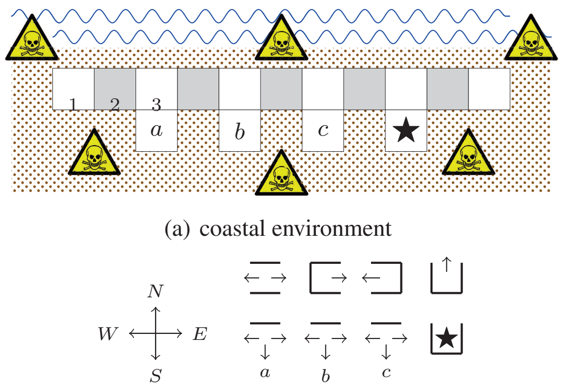
Abstract

In automated planning, action preconditions are boolean-valued formulas, which check whether a given action is feasible in a given state. While crucial for realistic applications where dangerous actions in some states must be discarded, preconditions have never been formally considered in POMDPs. One reason is that preconditions are defined over states whereas decisions depend on the current belief of the agent. Simply defining preconditions over beliefs is not sufficient because, as each belief is possibly defined over many states, there is no guarantee to prevent the agent from applying an infeasible damaging action. Augmenting the observation space with feasible actions does not help more, since the optimization process still maximizes the value of the current belief over all existing actions in the model. Thus, we propose an extension of the traditional POMDP model that, by means of an additional information step semantically different from standard observations, allows the agent to know the set of feasible actions before deciding the best action to apply. Without requiring a full knowledge of the current state, this extended model leads to a significant modification of the decision process, for which we provide a proved optimization scheme. We also compare the value and the execution paths of policies optimized either with the standard model or with our extended one, and show that our policies are always safe and gather more rewards at execution.

1 Introduction

In automated planning, preconditions are widely used to model environment properties required to perform an action. Preconditions are boolean-valued formulas that represent the definition domain of an action, i.e. the set of states on which this action can be applied [Ghallab *et al.*, 2004]. In real-world applications, securing this kind of guarantee is mandatory in order to protect a robotic agent against physical damage.

For example, consider an autonomous cost guard robot navigating along a cliff with abysses, as shown on Figure 1(a). This example is a slight variation of the Hallway problem, where surrounding walls are replaced by cliffs from which the robot can fall down. The agent can be in any square, and can apply 4 actions: *north, south, east, west*: Figure 1(b). The goal is to get to the star while being certain not to fall down a cliff: in the states near abysses, actions that might make the robot fall down *have to* be prohibited.



(a) coastal environment

(b) actions (c) observations

Figure 1: Coast guard POMDP problem.

In Markov Decision Processes (MDPs), recent efficient planners also rely on boolean-valued preconditions in order to produce policies that contain only feasible or desirable actions [Younes and Littman, 2003]. Former approaches used try to associate a high hand-tuned penalty to undesirable actions to discard them. This trick remain hazardous to tune and offer no guarantee that an infeasible action will not be inserted in the optimized policy. Furthermore, these approaches have been largely superseded by efficient algorithms that directly evaluate boolean-valued preconditions [Younes and Littman, 2003].

To our knowledge, the proper use of preconditions has never been adapted for Partially Observable Markov Decision processes (POMDPs), despite identical theoretical and practical needs. Research in POMDPs may still be more focused on improving the efficiency of general algorithms tackling the complexity of general POMDP models, rather than on real world applications. Technically speaking, the checking of preconditions is not straightforward when working in probabilistic partially observable domains, because the current state is not known precisely and replaced by a probability distribution over a set of states.

In this paper we first present the general POMDP framework in section 2 and discuss three workarounds used in order to discard undesirable actions, showing their weaknesses and drawbacks. In section 2.1 we show that the trick of tweaking costs of unsafe state-actions pairs is dependent on the chosen optimization algorithm and offers no guaranty that the actions will effectively be discarded. In section 2.2, we discuss the option of defining preconditions over the beliefs: we show that linking preconditions to beliefs may lead to incoherent policies. In section 2.3, we present a third option that would consist in augmenting the observations with the preconditions directly, which leads to potential tricky misfits in the semantics of observations and is still offering no guarantee.

The contribution of this paper is presented in section 3 and can be stated as follows: we propose a model, an adapted optimization scheme, compatible with many general POMDP algorithms, to properly take into account boolean-valued, state-based precondition formulas in POMDPs, which we claim is the right way to model and guarantee to ban undesired or unfeasible actions. In section 4, we show and experiment that this new model is fully compatible with different POMDP algorithms. We then present and discuss preliminary comparative results.

2 Background and workarounds

POMDP overview. Formally, a POMDP is a tuple $\langle S, A, \Omega, T, O, R, b_0 \rangle$ where S is the set of states; A the set of actions; Ω the set of observations; $T : S \times A \times S \rightarrow [0, 1]$ the transition function: $T(s_t, a, s_{t+1}) = p(s_{t+1}|a, s_t)$; $O : \Omega \times S \rightarrow [0, 1]$ the observation function: $O(o_t, s_t) = p(o_t|s_t)$; $R : S \times A \times S \rightarrow \mathbb{R}$ the reward function (associated with transitions), and b_0 the initial probability distribution over states. We denote Δ the (continuous) set of probability distributions over the states, named *belief space*.

At each time step t (assuming t discrete), the agent selects an action given the belief $b_t \in \Delta$, leading stochastically to a new state, then gets a noisy observation. The agent *belief* update, which is done using Bayes rule, depends on the action done and the observation gathered, and is a function of previous belief (s' follows state s):

$$b^o_a(s') = \frac{p(o|s') \sum_{s \in S} p(s'|s, a)b(s)}{\sum_{s \in S} \sum_{s'' \in S} p(o|s'')p(s''|s, a)b(s)} \qquad (1)$$

Abusing notation, we denote the belief $b_t(s) = Pr(s_t = s)$.

Solving a POMDP consists in finding a policy function $\pi : \Delta \rightarrow A$ that maximizes a performance criterion. The expected discounted reward from any initial belief $V^\pi(b) = E_\pi \left[\sum_{t=0}^{\infty} \gamma^t r(b_t, \pi(b_t)) \mid b_0 = b\right]$ is usually optimized. The value of an optimal policy π^* is defined by the optimal value function V^* that satisfies the Bellman optimality equation:

$$V^*(b) = \max_{a \in A} \left[\sum_{s \in S} r(s, a)b(s) + \gamma \sum_{o \in O} p(o|a, b)V^*(b^o_a) \right] \qquad (2)$$

This value function is piecewise linear and convex over the belief space [Sondik, 1971], so that at n^{th} optimization stage, the value function V_n can be parametrized as a set of hyperplanes over Δ named α-vectors. An α-vector and the associated action $a(\alpha^i_n)$ define a region of the belief space for which this vector maximizes V_n. Thus, the value of a belief b can be defined as $V_n(b) = \max_{\alpha^i_n \in V_n} b \cdot \alpha^i_n$. And the optimal policy at this step (b is the belief) is $\pi_n(b) = a(\alpha^b_n)$.

2.1 Tweaking costs in order to discard actions

Suppose the agent's belief is non zero only over three states 1, 2 and 3 of Figure 1(a). In these states, we must prevent the agent from moving towards cliffs. A simple, widely-used workaround consists in associating a high cost (ideally $-\infty$) for the infeasible actions: *north*, *west* and *south* in state 1, *north* and *south* in state 2, and *north* in state 3. As infinite values are not commonly modeled by computer libraries, one has to set a very low finite value in place of $-\infty$. Yet, the threshold of this value that ensures discarding infeasible actions from the optimized policy, actually depends on the unknown optimal value of states, and thus on problem's "regular" rewards and on the optimization criterion considered. In other words, for a given optimization criterion, we have to solve the problem before knowing the correct threshold of the high penalty associated to infeasible actions. For example, if reaching star gives the agent a reward of 1, and all safe actions have no cost; for $\gamma = 0.9$, even with a cost of 50, infeasible actions were executed 139 times over 500 simulations. Another workaround would consist in redefining low-level computational operations to deal with special values as infinity, but this is error-prone in complex optimization computations.

2.2 Belief-based preconditions

One might argue that a proper modeling of infeasible actions can be done by defining preconditions over a belief. But as actions are forbidden for real states, one has to define the banning of actions based on current belief.

On one hand, it is possible to have a *pessimistic approach*, consisting in forbidding an action as soon as there is a nonzero probability that it leads to a damage, but this may forbid useful actions: in our example, if the belief is non zero over states 1, 2 and 3 of Figure 1(a), the only safe action (i.e. does not move towards a cliff for certain) is *east* that, *by chance*, leads to the goal. Indeed, *east* would be the only action even if the star was on *a*, so we discard completely this approach.

On the other hand, we can try to prune minimally. For doing so, a simple strategy consists in computing a set of feasible actions based on the *support-states of the belief* (states where the probability $b(s) \neq 0$). Such a definition would *not* prevent for certain the robot to apply damaging actions. If we consider again the coast guard problem (Fig.1(a)) for a belief state over the states corresponding to numbers 1, 2 and 3. Following this definition, the set of feasible action is {*west,east,south*}, which is problematic with the state 1 where the agent should only perform *east*. The optimal action choice for this belief could lead to a wrong decision. More generally, if the belief is non zero on states where feasible actions are mutually exclusive (case of incoherent belief, for instance due to a wrong prior), this simple strategy will produce incoherent policies that apply infeasible actions at execution.

2.3 Augmenting observations with preconditions

In order to cope with such policy execution issue, another intuitive workaround consists in augmenting the observation space with the set of feasible actions, modeled as an additional observation variable, see Figure 1(c). Such extra information should make the robot choose only safe actions at execution; but we need to take a closer look here.

First, consider the initial belief b_0: if it is uniform, as usually done, then there is no way to forbid an unsafe action at first step; which means that there is a need for an initial observation, *not linked to a transition*, or that the initial belief has to be consistent with safe actions, and thus given by hand, which is always error-prone (especially, if b_0 gives a zero probability to too many states then belief updates might lead to inconsistencies when observations arrive).

Second, let us consider a belief that is non zero only over states 1, 2 and 3. The expected observations are $\sqsubset\!\!\rightarrow$, $\overline{\underline{\leftarrow\rightarrow}}$, $\overline{\leftarrow\underset{\downarrow a}{}\rightarrow}$ and $\overset{\uparrow}{\sqcup}$. Looking at the actions contained in these observations, Eq. 2 will be optimized over all actions ($\max_{a\in A}$). Therefore, even if observation $\overline{\underline{\leftarrow\rightarrow}}$ is received, the optimized action, which was defined based on the four possible observations, might still be *south*, maybe making the robot fall down the abyss. Indeed, action information must be provided *before* decision and not after. Thus, the aggregation operator (here max_A) needs to be changed; especially we want to filter out some actions from A set.

More precisely, the set of feasible actions has to be built in a clever way, by adding an additional information step, a priori quite similar to the standard observation step but semantically different from it, that informs the agent of the current feasible actions whatever its belief is. Indeed, the observations depend on the feasible actions so that both information must be clearly separated.

3 Planning in POMDPs with Preconditions

As discussed above, the only way to directly take into account boolean-valued preconditions in POMDPs (as opposed to indirect, unsafe tricks) is to add an additional information step, in order to restrict the belief to states where the set of current feasible actions is the same and thus, redefine the aggregation operator and optimize a policy with only feasible actions. This leads to a significant adaptation of both belief estimation and optimization processes of POMDPs, as detailed in this section. Before detailing such changes, we introduce the ideas and corresponding definitions and notations.

3.1 Action feasibility information

A precondition is a boolean-valued formula (or literal) that has to be true *for sure* if and only if an action is applicable in a given state. We note $\mathcal{A}_f(s)$ the set of *feasible actions* in a state s. The state-based preconditions are defined by means of a feasibility relation $\mathbb{I}$ saying that in a state s, an action a is allowed to be applied: $\mathbb{I}(a,s) = \mathbf{1}_{a\in\mathcal{A}_f(s)}$ where $\mathbf{1}_{cond}$ is 1 if the condition $cond$ is true, or 0 otherwise. $\mathbb{I}(a,s)$ can also be seen as the probability 1 or 0 of the applicability of an action conditioned to a state s, i.e. $\mathbb{I}(a,s) = Pr(a \in \mathcal{A}_f(s)|s_t = s)$.

Additional Information Step. Here, we suppose that the agent receives an additional information from the environment during policy execution, between the standard observation step and the execution step, that consists in the current set of feasible actions. This procedure is often performed in autonomous systems by special functionalities decoupled from planning, like execution control in [Ingrand *et al.*, 2007], in order to prevent the robot from damaging. Generally, the policy execution has to be controlled to insure that a safety plan will be conducted.

To take into account safety constraints on policy optimization and execution, POMDP models need to be extended. As we do not know in advance the set of feasible actions received from the environment, we need to plan for *all possible* feasible action sets $\{\widetilde{\mathcal{A}_f}^1, \cdots, \widetilde{\mathcal{A}_f}^j\}$, *independently from the belief*, with $j = 2^{|A|} - 1$ different sets of action combinations, where $|A|$ is the total number of actions. For the coast guard problem (Fig.1(a)), some possible sets would be: $\{$*west, east*$\}$, $\{$*west, south, east*$\}$, $\{$*east*$\}$ and $\{$*west*$\}$.

The key point is: if we have an action set information, the joint indicative function should be used in an *extra* belief update step before action execution. It allows us to optimize the value function only over feasible actions and corresponding observations. In other words, since observations depend on feasible actions, the observation step and the action feasibility one have to be separated, meaning that we need two different information steps. As we need to evaluate many sets of actions, we define the joint indicative function of a set of actions $\mathcal{U} \subset A$ over a state s as:

$$\mathbb{I}(\mathcal{U}, s) = \prod_{a_i\in\mathcal{U}} \mathbb{I}(a_i, s) \prod_{a_j\notin\mathcal{U}} (1 - \mathbb{I}(a_j, s)) \qquad (3)$$

We have directly $\mathbb{I}(\mathcal{A}_f(s), s)$=1. Moreover, it is interesting to note that $\mathbb{I}(\mathcal{U}, s) = Pr(\mathcal{A}_f(s) = \mathcal{U}|s)$, the 1 or 0 probability that a set of actions conditioned to a state s is equal to the actual set of feasible actions $\mathcal{A}_f(s)$. This simple equation allows to update the belief knowing the current set of feasible actions received. For instance, considering the coast guard problem of Fig. 1(a) with an initial uniformly distributed belief b, the agent receives $\mathcal{A}_f = \{$*west, east*$\}$ as action feasibility information, and projects the belief conditioned on $\mathcal{A}_f$, obtaining $\tilde{b}_{\mathcal{A}_f}$. Thus, the uncertainty will now be over the states in gray color of Fig. 1(a), where the set of feasible actions is the same. Thus, action optimization is constrained to feasible actions only. Note that the belief is not generally reduced to a single state after this additional information step, still keeping the agent's observability partial.

Using support-states information. It is possible to include in our approach the minimal pruning described in section 2.2. It will reduce the support of the belief, that should converge faster towards the hidden state. Yet, this approach can lead to an empty action choice if the belief is initially equal to zero on the hidden state (discussed below). The set of feasible actions knowing belief b is:

$$\mathcal{A}_f(b) = \{a \in A | \exists s \in S, b(s) \neq 0 \wedge \mathbb{I}(a, s) = 1\} \qquad (4)$$

Now, suppose the robot receives the set of feasible actions $\widetilde{\mathcal{A}_f}^i$ at a given time t. With this definition, we can restrain the set of feasible actions to $\mathcal{A}_f^i = \mathcal{A}_f(b) \cap \widetilde{\mathcal{A}_f}^i$. The corresponding optimization scheme means that we optimize the value over all such set intersections, where many of them are hopefully empty or give rise to smaller sets compared to the approach that does not use the support-states information. We note $\{\mathcal{A}_f^1, \cdots, \mathcal{A}_f^n\}$ the set of possible sets of feasible actions, i.e. all combinations of actions intersected with $\mathcal{A}_f(b)$. We have $n = 2^{|\mathcal{A}_f(b)|} - 1$. Without loss of generality, if the additional action feasibility information is not available, the intersection $\mathcal{A}_f(b) \cap \widetilde{\mathcal{A}_f}^i$ is not defined and $\mathcal{A}_f^i = \mathcal{A}_f(b)$.

Three cases can give an empty intersection. The first one, $\mathcal{A}_f(b) = \emptyset$, is impossible, because $\sum_s b(s) = 1$. The second one, $\widetilde{\mathcal{A}_f}^i = \emptyset$, is also impossible, because it exists at least one action per set when combinations are generated. Thirdly, if $\mathcal{A}_f(b)$ and $\widetilde{\mathcal{A}_f}^i$ are mutually exclusive, then there is an inconsistency between the belief and the external information; we will not take into account this case.

3.2 Belief update w.r.t. action feasibility

This new information available to the agent, leads to two belief update stages:

1. A projection of a b is made for each possible set of feasible actions $\mathcal{A}_f^i$, resulting in $\tilde{b}_{\mathcal{A}_f^i}$.

$$\begin{aligned}\tilde{b}_t(s)_{\mathcal{A}_f^i} &= Pr(s_t = s|\mathcal{A}_f^i) = \frac{Pr(\mathcal{A}_f^i|s_t = s)Pr(s_t = s)}{Pr(\mathcal{A}_f^i)} \\ \tilde{b}_t(s)_{\mathcal{A}_f^i} &= \frac{\mathbb{I}(\mathcal{A}_f|s_t = s)b_t(s)}{\sum_{s''}\mathbb{I}(\mathcal{A}_f|s_t = s'')b_t(s'')} \qquad (5)\end{aligned}$$

If the belief is incoherent with the set of feasible actions the denominator becomes zero, what invalidates Bayes' rule. This can happens if $\mathcal{A}_f^i = \widetilde{\mathcal{A}_f}^i$, i.e. when $\forall a \in \widetilde{\mathcal{A}_f}^i,\ a \notin \mathcal{A}_f(b)$. In such cases, we can discard the current belief information and simply reset the belief.

2. An action $a \in \mathcal{A}_f^i$ is chosen, and observation o is received, bringing the agent to a belief b_a^o. The rest of the update of $\tilde{b}$ is identical to the standard model given that the chosen action belongs to $\mathcal{A}_f$, and b is substituted by $\tilde{b}$ in Eq. 1:

$$b^o_{a\in\mathcal{A}_f}(s') = \frac{p(o|s')\sum_{s\in S}p(s'|s,a)\tilde{b}(s)}{\sum_{s\in S}\sum_{s''\in S}p(o|s'')p(s''|s,a)\tilde{b}(s)} \qquad (6)$$

The additional action feasibility step does not invalidate the assumption that the belief probability distribution is a complete state information, because it still is a markovian process. Furthermore, at execution time, the agent has to make decisions based on $\tilde{b}$; for example at steps k and $k+1$ of Fig. 2, after the feasibility information step. Thus, the optimization algorithm has to implement backups over $\tilde{b}$. In cases where action feasibility information is unavailable, the scheme is reduced to the standard belief update provided that $a \in \mathcal{A}_f(b)$.

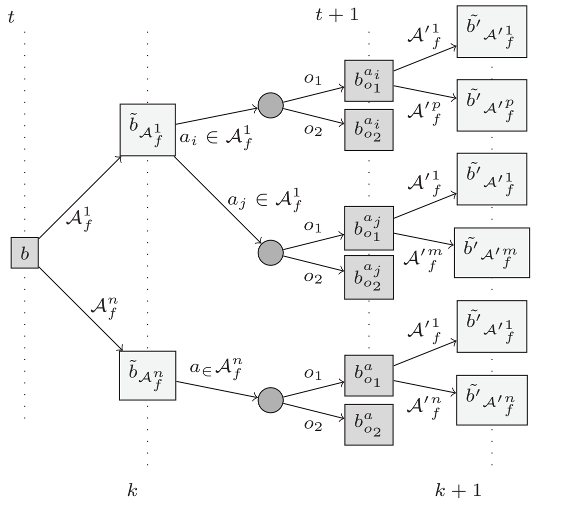


Figure 2: New belief update scheme.

3.3 Optimization: extending backup operator

As an information step is added to the belief update, one has to adapt Bellman's equation in order to average the expected value of a belief $\tilde{b}$ over the intermediate beliefs $b^o_{a\in\mathcal{A}_f}$:

$$V_{k+1}(\tilde{b}_{\mathcal{A}_f}) = \max_{a\in\mathcal{A}_f}\{r(\tilde{b},a) + \textstyle\sum_o p(o|a,\tilde{b})V_n(b^o_{a\in\mathcal{A}_f})\} \quad (7)$$

where $\mathcal{A}_f$ represents the set of feasible actions associated with the belief $\tilde{b}$. Note that the value $V_{k+1}(\tilde{b}_{\mathcal{A}_f})$ is linear in $\tilde{b}$, but also depends on the value of the future belief $b^o_{a\in\mathcal{A}_f}$. Computing $V_n(b^o_{a\in\mathcal{A}_f})$ is not trivial, because it depends on next $\tilde{b}_{\mathcal{A}'_f}$. We compute $V_n(b^o_{a\in\mathcal{A}_f})$ as an average:

$$V_n(b) = \textstyle\sum_{i=1}^{|C(b)|} Pr(\mathcal{A}_f^i)V_k(\tilde{b}_{\mathcal{A}_f^i}(b)) \qquad (8)$$

This leads to a non-linear dependence, since the number of successors of b, denoted $|C(b)|$, depends on the number of possible sets of feasible actions. The average value of successors of b holds only locally, in other words, we know the number $|C(b)|$ of successors for a given b only. Thus, we can locally approximate the value function with α-vectors: using the fact that Eq. 7 is linear in $\tilde{b}$, we redefine the backup operator presented in [Pineau *et al.*, 2003]. So, $\forall a \in \mathcal{A}_f$, $\forall o \in \Omega$ and $\forall \bar{\alpha}_i \in V_n$, with V_n of Eq. 8:

$$\begin{aligned}\Gamma^{a,*} &\leftarrow \alpha^{a,*}(s) = R(s,a)\mathbb{I}(a,s) \\ \Gamma^{a,o} &\leftarrow \alpha^{a,o}(s) = \gamma\textstyle\sum_{s'\in S}p(s'|s,a)\mathbb{I}(a,s)p(o|s')\bar{\alpha}_i(s')\end{aligned}$$

The α-vector associated with $a \in \mathcal{A}_f$ is constructed as:

$$\Gamma^a_{\tilde{b}} = \Gamma^{a,*} + \textstyle\sum_{o\in\Omega}\arg\max_{\alpha\in\Gamma^{a,o}}(\alpha\cdot\tilde{b}) \qquad (9)$$

We use the α-vector maximizing $\tilde{b}$ to compute V_{k+1}:

$$V_{k+1} \leftarrow \underset{\Gamma^a_{\tilde{b}},\forall a\in\mathcal{A}_f}{\arg\max}(\Gamma^a_{\tilde{b}}\cdot\tilde{b}) \qquad (10)$$

As V_k can be represented by α-vectors, Eq. 8 becomes:

$$V_n(b) = \textstyle\sum_{i=1}^{|C(b)|}Pr(\mathcal{A}_f^i)\arg\max_{\tilde{\alpha}\in V_k}\tilde{\alpha}\cdot\tilde{b}(b) \qquad (11)$$

and the relationship between $\tilde{b}$ and b is given by Eq. 5, which is denoted in a vector form as $\tilde{b} = \frac{\mathbb{III}(\mathcal{A}_f)b}{Pr(\mathcal{A}_f)}$, where $\mathbb{III}(\mathcal{A}_f)$ is a diagonal matrix $S \times S$ with terms $\mathbb{III}_{ii} = \mathbb{I}(\mathcal{A}_f, s_i)$. So:

$$\begin{aligned}V_n(b) &= \textstyle\sum_{i=1}^{|C(b)|}Pr(\mathcal{A}_f^i)\arg\max_{\tilde{\alpha}\in V_k}\tilde{\alpha}\cdot\frac{\mathbb{III}(\mathcal{A}_f^i)b}{Pr(\mathcal{A}_f^i)} \quad (12) \\ V_n(b) &= \textstyle\sum_{i=1}^{|C(b)|}\arg\max_{\tilde{\alpha}\in V_k}\tilde{\alpha}\cdot\mathbb{III}(\mathcal{A}_f^i)b \quad (13)\end{aligned}$$

and finally, V_n is constructed as:

$$V_n \leftarrow \textstyle\sum_{i=1}^{|C(b)|}\arg\max_{\tilde{\alpha}\in V_k}\tilde{\alpha}\cdot\mathbb{III}(\mathcal{A}_f^i) \qquad (14)$$

$\mathbb{III}(\mathcal{A}_f^i)$ works as a mask over b, in order to construct an α-vector with values defined only for states where $\mathcal{A}_f^i$ holds. The value of b is locally approximated by averaging the α-vectors of its successors. If action feasibility information is available, the backup operator boils down to the standard one [Pineau *et al.*, 2003], but α-vector projections and value calculations are restrained to actions a such that $a \in \mathcal{A}_f(b)$.

Some advantages of this new backup operator are: (1) We take into account the actions that belong to a set of feasible actions for each belief $\tilde{b}$, evaluations for *all action* of the model are no more made. (2) The new α-vectors are sparse, which is exploited in computations. This sparsity is due to the (un-)feasibility of an action over states; α-vectors are hyperplanes defined on the belief simplex. So, if for a state s an action is forbidden, the value associated with this action is not defined.

Theorem: Contraction of the new backup operator. *Let* $\gamma < 1$. *The new backup operator defined as:*

$$\mathcal{L}V(\tilde{b}) = \max_{a\in\mathcal{A}_f} r(\tilde{b},a) + \gamma\sum_{o\in\Omega}p(o|a,\tilde{b})\sum_{i=1}^{C(b_a^o)}Pr(\mathcal{A}_f^i)V(\tilde{b}^{a,o}_{\mathcal{A}_f^i})$$

is a contraction over $\mathcal{V}$*, the value function space.*

Proof. Let $V \in \mathcal{V}$, $U \in \mathcal{V}$, and $\tilde{b} \in \Delta$, where $||\tilde{b}|| = \sum_{s \in S} |\tilde{b}(s)| = 1$. We suppose $\mathcal{L}V(\tilde{b}) \geq \mathcal{L}U(\tilde{b})$, and

$$a^* = \arg\max_{a \in \mathcal{A}_f} \{r(\tilde{b}, a) + \gamma \sum_{o \in \Omega} p(o|a, \tilde{b}) \sum_{i=1}^{C(b_a^o)} Pr(\mathcal{A}_f^i) V(\tilde{b}^{a,o}_{\mathcal{A}_f^i})\}$$

we have: $|\mathcal{L}V(\tilde{b}) - \mathcal{L}U(\tilde{b})| = \mathcal{L}V(\tilde{b}) - \mathcal{L}U(\tilde{b})$

$$\leq \gamma \sum_{o \in \Omega} p(o|a^*, \tilde{b}) \sum_{i=1}^{C(b_{a^*}^o)} Pr(\mathcal{A}_f^i) ||V - U|| \leq \gamma ||V - U||$$

so: $||\mathcal{L}V - \mathcal{L}U|| = \max_{||\tilde{b}||=1} |\mathcal{L}V(\tilde{b}) - \mathcal{L}U(\tilde{b})| \leq \gamma ||V - U||$

As our new backup operator satisfies the contraction property, value iteration scheme converges. □

On the complexity. In our extended POMDP model, all possible subsets of actions must be assessed at each time step in planning, so that, in the worst case, the complexity of POMDP algorithms is multiplied by $2^{|A|} - 1$, where '-1' represents the empty action set. According to us, this is the computational price to guarantee that no infeasible actions are applied at execution in a POMDP context. But this is only the worst case, since in practice as highlighted by our experiments (see next section), the belief state is more pruned before each decision step, what reduces the number of alpha-vectors generated in most cases.

4 Experiments

Recent POMDP algorithms, like PBVI [Pineau *et al.*, 2003], Perseus [Spaan and Vlassis, 2004], HSVI2 [Smith and Simmons, 2005], SARSOP [Kurniawati *et al.*, 2008] approximate the value function using a finite set B of beliefs, where $B \subset \Delta$. These algorithms implement various heuristics for exploring the belief space, and update the value function V, defined as a set of α-vectors, for each b explored. So, V is constrained to contain at most $|B|$ α-vector components. The main difference between them is the way they explore the belief space and/or maintain the upper and lower bounds.

4.1 Experimental setup

In order to validate our approach, we implemented PCVI – *PreCondition Value Iteration*, a point-based POMDP algorithm based on PBVI. PCVI works on a finite set of beliefs $\tilde{B} = \tilde{b}_0, ...\tilde{b}_k$ and uses the new belief update step and the new backup operator allowing to handle actions preconditions, either conditioned on external information or not. PCVI, as PBVI and Perseus, explores the belief state space by stochastic simulations, and this exploration depends on availability of external information. Both approaches presented on section 3 can be chosen. Without loss of generality, if no external information is available, $\mathcal{A}_f$ is defined as the minimal pruning described on support-states approach (section 2.2 and Eq. 4). Note that any POMDP algorithm for the standard model can be generalized to this decision scheme in the same way.

Here, we compare policies optimized with the standard POMDP model to policies optimized with our additional action feasibility information step, in terms of mission achievement and safety. Any mission where a forbidden action is chosen is considered as failed. Even if *we are not interested in comparing algorithm performances but models*, we highlight that our additional information step does not increase computation times nor decreases rewards gathered.

We conducted experiments on 8 well-known POMDP problems: *maze4x3*, *maze4x5x2*, *Hallway*, *Hallway2*, *aircraft* and *iff* [Cassandra, 1998], *tiger-grid* and *RockSample4x4* [Smith and Simmons, 2005], with PBVI, HSVI2, and PCVI algorithms. In navigation problems, the agent is forbidden to move towards a wall. In rockSample it is forbidden to get rocks where there is no. On aircraft problems it is forbidden to use the active radar when a target is close from the base.

In usual algorithms, like PBVI and HSVI2, action feasibility is not formally taken into account: instead, we associate a high penalty to state-actions pairs (s, a) so that choosing action a in state s is discouraged. Even if we add an additional observation variable representing feasible actions, standard algorithms cannot deal with them correctly (see section 2.3).

4.2 Comparative results

Table 1 summarizes performances averaged over 500 policy simulations. We can see that, for most problems, rewards gathered are higher for PCVI with or without the support-states information. This is because our algorithm does not use infeasible actions at all and so is not "punished" with a bad reward (we directly compute a policy that *guarantees* that an infeasible action is *never* chosen). Moreover, the number of α-vectors is generally lower with our approach.

Fig. 3 shows more results for *maze4x3* and *hallway[1,2]* problems. It compares our model with the standard one for three criteria: (1) number of times that infeasible actions are chosen; (2) percentage of goal achievement; (3) number of steps to reach the goal. These criteria are measured over 500 policy executions. Criterion (1) is the main one for our study. Criteria (2) and (3) are presented to highlight that our extended POMDP model is competitive with, if not better than, the standard model for optimization-related criteria. For the *maze4x3* problem, even with the highest penalty of 50, infeasible actions are applied with the standard model. On *hallway* problems, the use of infeasible actions decreases when penalties increase. Yet, the penalty threshold that makes the agent avoid infeasible actions is not known in advance.

Our approach ensures that no infeasible action is chosen independently from any hand-tuned penalty. Looking at criterion (2), one can see that for large penalties, the number of successful simulations decreases for PBVI policies, that prioritizes the non use of infeasible actions, contrary to HSVI2 that prefers to relax actions to reach the goal.

Our approach reaches the goal almost 100% of the time, even if this criterion is not targeted by our model. Regarding criterion (3), in the *maze4x3* problem, our new belief update step significantly reduces the support of agent's belief, so that the average number of steps to reach the goal also is reduced. In *hallway* problems, the average number of steps is slightly higher than the average of the others, but this average is based on successfully simulations, and this explains the fact that PBVI policy has a good steps average and a bad success percentage compared to HSVI2 and PCVI policies.

5 Conclusion and Future Work

In this paper, we propose a new POMDP framework using boolean-valued preconditions to guarantee that the optimized policy contains only feasible actions. This requires to adapt the decision scheme, and so the optimization criterion and

	maze4x3 (11s/4a/6o) p.1 h.50			**maze4x5x2** (39s/4a/4o) p.1 h.100			**hallway** (60s/5a/21o) p.1 h.250			**hallway2** (92s/5a/17o) p.1 h.250		
	Reward	Time(s)	$\|\Gamma\|$	Reward	Time(s)	$\|\Gamma\|$	Reward	Time(s)	$\|\Gamma\|$	Reward	Time(s)	$\|\Gamma\|$
PBVI	0.089	5.913	25	0.061	191.86	166	0.470	1747	251	0.204	4489	491
HSVI2	0.015	65.25	260	-0.077	14.29	1132	0.480	2729	935	0.258	313.3	2047
PCVI-	0.072	8.344	27	-0.064	373.4	201	0.482	3140	273	0.202	6236	482
PCVI	0.540	0.880	3	0.634	43.60	24	0.516	1305	259	0.310	973.7	278
PCVI+	0.552	2.757	7	0.470	53.77	23	0.541	1670	262	0.344	977.5	291

	aircraft (12s/6a/5o) p.1 h.50			**iff** (104s/4a/22o) p.30 h.100			**tiger-grid** (36s/5a/17o) p.10			**rockSample4x4** (257s/9a/2o) p.100		
	Reward	Time(s)	$\|\Gamma\|$	Reward	Time(s)	$\|\Gamma\|$	Reward	Time(s)	$\|\Gamma\|$	Reward	Time(s)	$\|\Gamma\|$
PBVI	12.84	114.2	26	7.262	5839	420	0.579	2343	418	16.29	15893	65
HSVI2	13.26	0.065	3	-7.163	2839	13983	0.532	5432	14168	18.15	1.059	287
PCVI-	12.13	142.8	23	7.746	10322	421	0.617	2295	399	16.77	5683	111
PCVI	14.30	129.4	35	7.257	4093	244	0.625	1046	286	16.36	8423	94
PCVI+	15.02	120.3	22	8.425	7480	344	0.632	841.6	220	15.81	26922	76

p. = penalty, h. = horizon, $|\Gamma|$ = number of α-vectors, PCVI- = support-states, PCVI = additional information, PCVI+ = mixed

Table 1: Multi-algorithm performance comparison.

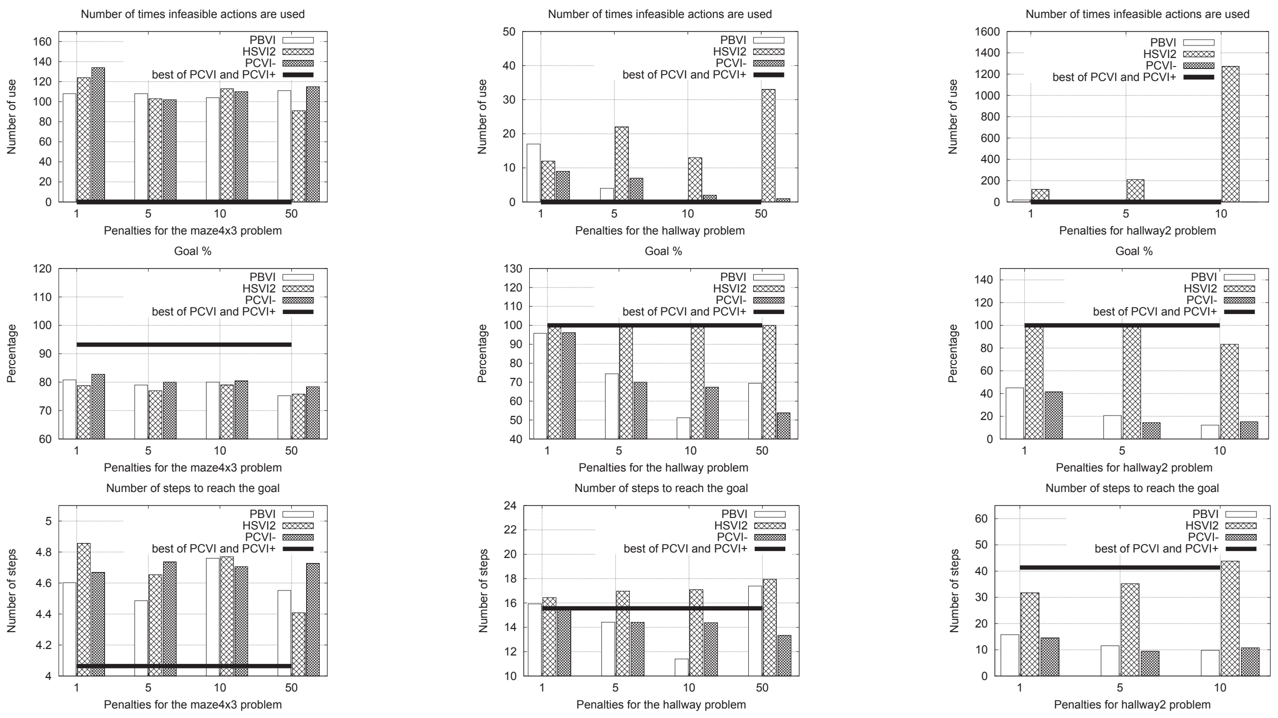


Figure 3: Navigation problems performance ; PCVI- = support-states, PCVI = additional information, PCVI+ = mixed

corresponding value update equation, for which we demonstrate the contraction property.

The implementation results, with a modified version of PBVI named PCVI, confirm that our approach guarantees to never apply forbidden actions at execution, independently from any tricky hand-tuning of penalties. Furthermore, this approach shows competitive results in terms of performances: useful pruning on constrained problems, good rate of goal achievement, higher rewards and value.

We intend to extend the approach and study other aggregation operators for the value function of a belief b as a function of the value of its successor $\tilde{b}$. We also intend to extend the approach with efficient state-of-the-art heuristic search algorithms : modified HSVI2 or SARSOP for instance, which will require to adapt the belief update scheme, backup operator and heuristics for the boolean-valued precondition functions.

References

[Cassandra, 1998] A.R. Cassandra. *Exact and approximate algorithms for partially observable Markov decision processes*. PhD thesis, Brown University Providence, RI, USA, 1998.

[Ghallab *et al.*, 2004] M. Ghallab, D.S. Nau, and P. Traverso. *Automated Planning: theory and practice*. Morgan Kaufmann, 2004.

[Ingrand *et al.*, 2007] F. Ingrand, S. Lacroix, S. Lemai-Chenevier, and F. Py. Decisional autonomy of planetary rovers. *Journal of Field Robotics*, 24(7):559–580, 2007.

[Kurniawati *et al.*, 2008] H. Kurniawati, D. Hsu, and W.S. Lee. SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces. In *Proc. RSS*, 2008.

[Pineau *et al.*, 2003] J. Pineau, G. Gordon, and S. Thrun. Point-based value iteration: An anytime algorithm for POMDPs. In *Proc. of IJCAI*, 2003.

[Smith and Simmons, 2005] Trey Smith and Reid G. Simmons. Point-based POMDP algorithms: Improved analysis and implementation. In *Proc. UAI*, 2005.

[Sondik, 1971] E.J. Sondik. The optimal control of partially observable Markov processes, 1971.

[Spaan and Vlassis, 2004] M.T.J. Spaan and N. Vlassis. A point-based POMDP algorithm for robot planning. In *ICRA*, 2004.

[Younes and Littman, 2003] H.L.S. Younes and M.L. Littman. PPDDL1.0: An extension to PDDL for expressing planning domains with probabilistic effects. In *In Proc. of ICAPS*, 2003.