Heuristics for Planning under Partial Observability with Sensing Actions

Guy Shani and **Ronen Brafman** and **Shlomi Maliah**
Ben Gurion University

Erez Karpas
Technion

Abstract

In planning under partial observability with sensing actions (PPOS) problems, the solution progresses from one sensing action to another, until sufficient information is gathered and the goal can be reached. In between sensing actions, one can use classical planning to derive the path to the next sensing action. We suggest an online algorithm that repeatedly selects the next sensing action to execute, and plans to achieve it in a classical setting. Our algorithm avoids the difficulty in representing and updating a belief space. Our heuristic uses landmarks, and we explain how landmarks can be computed over a relaxation of the PPOS problem. We compare our Heuristic Contingent Planner (HCP) to state-of-the-art, translation-based online contingent planners, and show how it solves many problems much faster than previous approaches.

Introduction

Agents acting under partial observability must acquire information about the true state of the world using sensing actions to achieve their goals. Such problems can be modeled using contingent planning, where action effects may be conditioned on some unknown world features. Contingent planning is difficult because the plan must branch given different sensor values, resulting in potentially large plan trees.

Currently, there are two popular techniques for generating contingent plans. Both approaches can be used in an offline scenario – where the entire plan tree is generated offline, or in an online scenario — where only the branch corresponding to the online observations is generated incrementally.

The first technique builds on a compilation method for conformant planning (Palacios and Geffner, 2009), translating the contingent planning problem, into a classical planning problem that "reasons" about the agent's state of knowledge. Solutions to this classical problem correspond to complete or partial contingent plans (Albore, Palacios, and Geffner, 2009; Shani and Brafman, 2011; Bonet and Geffner, 2011a). A major difficulty with this approach is the size of the generated classical planning problems and the time required for their solution.

The second technique directly searches in belief space (Bonet and Geffner, 2000; Hoffmann and Brafman, 2005; To, Son, and Pontelli, 2011). As the effect of sensing actions is non-deterministic, the search can be modeled as an AND/OR tree, and various search heuristics can be applied. The major difficulty with this approach is to compactly represent and efficiently update the belief state, and currently no known method works well in all domains (To, Pontelli, and Son, 2011). Generating good heuristics estimates in belief space is also a challenge.

This paper describes a new, online method that overcomes the weaknesses of both methods. We solve a sequence of simple classical planning problems defined on the original state space. Thus, we avoid reasoning about the agent's knowledge, and maintaining and updating its belief state. Our method currently focuses on contingent planning domains in which sensing actions do not alter the state of the world — a condition which is true in all current contingent planning benchmarks. In such domains, every branch of a contingent plan contains a sequence of sensing actions. Between every two sensing actions, there is a sequence of non-sensing actions. This is well known, and most online contingent planners leverage it, whether explicitly or implicitly.

The key insight in our method, though, is that the sequence of non-sensing actions in between every two sensing actions can be viewed as a classical planning problem defined over the original (in fact, even slightly reduced) state space of the problem, rather than the belief space. Thus, if at each stage we are given the next sensing action to perform by an oracle, we can quickly plan to achieve its preconditions using a suitable classical planner.

Given the agent's current state, there may be multiple reachable sensing actions. Thus, we require a method for selecting among them — an oracle. Indeed, technically, our main contribution is the development of a method for approximating the value of information of every achievable sensing action using an estimate of the number of landmarks that can be achieved following the execution of each sensing action. This requires, in turn, adjusting landmark generation techniques to the setting of partial observability and sensing.

Our Heuristic Contingent Planner (HCP) first computes a set of landmarks over a specially designed relaxation of the planning problem. Next, at each step, we compute the set of reachable sensing actions and estimate their value of information. We use then a classical planner to generate a plan from the current state to the seemingly best sensing action. We repeat this process until we reach the goal.

We empirically evaluate HCP on a set of contingent planning benchmarks. Our experiments show that HCP is much faster than the state-of-the-art contingent planners SDR (Brafman and Shani, 2012b) and CLG (Albore, Palacios, and Geffner, 2009), on domains with structured belief spaces, where certain conditions that we explain hold, and good landmarks can be discovered.

Partially Observable Contingent Planning

Partially observable contingent planning problems are characterized by uncertainty about the initial state of the world, partial observability, and the existence of sensing actions. Actions may be non-deterministic, but much of the literature focuses on deterministic actions, and in this paper we will assume deterministic actions, too.

Problem Definition

A contingent planning problem is a quadruple: $\pi = \langle P, A, \varphi_I, G \rangle$. P is a set of propositions, A is a set of actions, φ_I is a formula over P that describes the set of possible initial states, and $G \subset P$ is the goal propositions. We often abuse notation, treating a set of literals as a conjunction of the literals in the set, as well as an assignment of the propositions in it. For example, $\{p, \neg q\}$ will also be treated as $p \wedge \neg q$ and as an assignment of *true* to p and *false* to q.

A state of the world, s, assigns a truth value to all elements of P. A *belief-state* is a set of possible states, and the initial belief state, $b_I = \{s : s \models \varphi_I\}$ defines the set of states that are possible initially. An action $a \in A$ is a three-tuple, $\{$*pre*(a),*effects*(a),*obs*$(a)\}$. *pre*(a) is a set of literals denoting the action's preconditions. *effects*(a) is a set of pairs (c, e) denoting conditional effects, where c is a set (conjunction) of literals and e is a single literal. Finally, *obs*(a) is a set of propositions, denoting those propositions whose value is observed when a is executed. We assume that a is well defined, that is, if $(c, e) \in$ *effects*(a) then $c \wedge$*pre*(a) is consistent, and that if both $(c, e), (c', e') \in$ *effects*(a) and $s \models c \wedge c'$ for some state s then $e \wedge e'$ is consistent. In current benchmark problems, either the set *effects* or the set *obs* are empty. That is, actions either alter the state of the world but provide no information, or they are pure sensing actions that do not alter the state of the world, but this is not a mandatory limitation.

We use $a(s)$ to denote the state that is obtained when a is executed in state s. If s does not satisfy all literals in *pre*(a), then $a(s)$ is undefined. Otherwise, $a(s)$ assigns to each proposition p the same value as s, unless there exists a pair $(c, e) \in$ *effects*(a) such that $s \models c$ and e assigns p a different value than s. Observations affect the agent's belief state. We assume throughout that all observations are deterministic and accurate, and reflect the state of the world *prior* to the execution of the action. It is possible to have observation reflect the post-action state, at the price of slightly more complicated notation. Thus, if $p \in$*obs*(a) then following the execution of a, the agent will observe p if p holds now, and otherwise it will observe $\neg p$. Thus, if s is the true state of the world, and b is the current belief state of the agent, then $b_{a,o}$, is the belief state following the execution of a and observing o. The new belief state corresponds to the progression through a of all states in b where o is observed.

A complete plan for a contingent planning problem can be described as an tree $\tau = (N, E)$. The nodes, N, are labeled with actions, and the edges, E, are labeled with observations. A node labeled by an action with no observations has a single child, and the edge leading to it is labeled by the null observation *true*. Otherwise, each node has one child for each possible observation value. The edge leading to this child is labeled by the corresponding observation. τ is a *solution plan* (or *plan*) for π if $\tau(s) \models G$ for every $s \in b_I$.

We illustrate these ideas using a 4×4 Wumpus domain (Albore, Palacios, and Geffner, 2009), which will serve as our running example. Figure 1 illustrates this domain, where an agent is located on a 4×4 grid. The agent can move in all four directions, and if moving into a wall, it remains in place. The agent initially is in the low-left corner and must reach the top-right corner. There are two monsters called Wumpuses hidden along the grid diagonal, the agent knows that Wumpus 1 can be at location 3,2 or 2,3, and Wumpus 2 can be at location 4,3 or 3,4. Thus the possible states are: $\{wat(3,2) \wedge wat(4,3), wat(3,2) \wedge wat(3,4), wat(2,3) \wedge wat(4,3), orwat(2,3) \wedge wat(3,4)\}$. The stench of a Wumpus carries to all adjacent locations, and the agent can observe the stench in order to deduce the whereabouts of the Wumpuses.

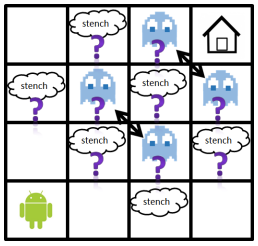

Figure 1: The 4×4 Wumpus domain

Landmarks in Classical Planning

In general, a landmark Φ for a classical planning task $\Pi = \langle P, A, I, G \rangle$ is a logical formula over the facts P, which must be satisfied at some state along every solution of Π (Hoffmann, Porteous, and Sebastia, 2004). As in most literature dealing with landmarks, we will restrict our attention to landmarks that are simple disjunctions or conjunctions over facts. Each planning task has some trivial landmark, consisting of all goal facts, and all facts in the initial state. In the Wumpus problem described above, e.g., these trivial landmarks are the goal, *at-4-4*, and the initial state fact *at-1-1*.

Another related notion is that of orderings between landmarks. Several types of orderings have been defined (Hoffmann, Porteous, and Sebastia, 2004). We use the most general type of ordering (that is, the one with the weakest condition), called reasonable-ordering, denoted $\Phi \rightarrow \Psi$. Intuitively, this implies that a "reasonable" plan will achieve Φ before Ψ; the exact definitions of the different orderings are not important in this context.

Although it is PSPACE-hard even to check whether a given fact is a landmark or not, there are several efficient

algorithms which return a set of landmarks and orderings (Hoffmann, Porteous, and Sebastia, 2004; Zhu and Givan, 2003; Richter and Westphal, 2010; Keyder, Richter, and Helmert, 2010). These algorithms all exploit the principle that, if some fact $p \in P$ is a landmark that is not true in the initial state, and all actions which achieve p have some fact $q \in P$ as a precondition, then q is also a landmark. Furthermore, q must be achieved before p, and so we also have the ordering $q \rightarrow p$. If there is no common precondition, we can still find a set of facts which occur in the preconditions of all actions, and use them as a disjunctive landmark. For example, the landmark *at-4-4* has two achievers, going up from *at-4-3*, and going right from *at-3-4*. These actions do not have a common precondition, but we can still infer that *at-4-3* $\vee$ *at-3-4* is a landmark, which is ordered before *at-4-4*.

Originally, landmarks were used as subgoals (Hoffmann, Porteous, and Sebastia, 2004), guiding a base planner inside a control loop. At each iteration of the loop, the set of landmarks which could be achieved — that is, the landmarks which do not have an unachieved landmark that is ordered before them — are passed to the base planner as a disjunctive goal. The base planner then returns a plan which achieves one or more of these landmarks, the landmarks which have been achieved are marked, and the control loop continues planning from the state which was reached. Although this technique has been shown to speed up planning significantly, it can not guarantee the optimality of the solution found, nor even the completeness of the overall planning process. This is because the base planner might reach a state which is a dead-end for the planning task Π, and the control loop does not backtrack.

More recently, the number of landmarks which are yet to be achieved was used as a path-dependent heuristic in the LAMA planner (Richter and Westphal, 2010), winner in the sequential satisfying track in IPC-2008 and IPC-2011. This is an inadmissible heuristic estimate, because an action might achieve more than one landmark. If optimal planning is of interest, it is possible to derive an admissible landmarks-based heuristic by performing an action cost partitioning over the landmarks (Karpas and Domshlak, 2009).

Landmark Detection under Partial Observability

Adapting landmark detection to PPOS, we must be able to handle uncertainty and sensing. In principle, we could run classical landmark detection in belief space or in the pseudo belief space of the translation approach, both of which are potentially exponentially larger. To be useful, however, landmark detection must be efficient and informative.

One of the main goals and contributions of this work is to use classical techniques on the original state space when possible, while planning in a partially observable domain. To reduce the detection cost, as in classical planning, we run landmark detection on a relaxed problem. We use the popular delete-relaxation approach in which negative effects of actions are ignored.

In addition to ignoring delete-effects, we augment the domain with additional artificial actions that help us deal with uncertainty and observation. As the value of some propositions is unknowns, there is typically no path from the initial state to a goal state that does not involve observations and deductions from these observations concerning the set of possible states. The actions that we add help us bridge this gap to a certain extent, without requiring us to represent the agent's belief state.

Our relaxation is motivated by features of current benchmarks, as we explain below. In that sense, it is not general and may fail to produce meaningful landmarks when certain conditions do not apply. That being said, we demonstrate how our relatively simple relaxation is highly useful in a large set of benchmarks. Like other landmark detection algorithm, our approach is sound, in that every landmark that is detected is indeed a landmark of the real domain, but incomplete, in that many landmarks may go undetected. We skip the soundness proof due to the lack of space.

Conditional Effects Removal

We break down each action into actions with unary effects, by replacing each action $a \in A$ with conditional effects $\{(c_i, e_i) : i = 1..k\}$, with k actions, such that $pre(a_{(c_i,e_i)}) = pre(a) \wedge c_i$ and $\mathit{effects}(a_{(c_i,e_i)}) = \mathit{effects}(a) \wedge e_i$.

In general such a compilation is unsound, because several conditions may apply together at a given state, while we allow only one condition per action. As we are computing a heuristic, though, and since we will apply these actions in a delete-relaxation, forward-search manner, we will allow the application of multiple actions concurrently (Blum and Furst, 1997). Thus, all conditions that apply will be executed simultaneously at the same phase of the forward search.

Reasoning over the Initial Uncertainty

In many cases the uncertainty over some sets of propositions is encoded into the initial belief state, and remains unchanged throughout the execution of the solution (Bonet and Geffner, 2011b). In the Wumpus domain, e.g., the monsters remain at a given location and never move, and the stench around them is unchanged. For such sets of propositions, we can create reasoning actions that, upon detecting the value of some propositions of that set, reason about the rest.

The detection of these sets of propositions is simple, as propositions whose value is constant do not appear in the effects of any action. We find clauses of the initial belief formula that contain only constant propositions, and use these to create reasoning actions. We assume here that the initial state formula is expressed as a conjunction of either simple disjunctions of literals, or xor (so called "one-of") clauses.

For each such clause c containing only unchanged propositions we create a set of "reasoning" actions A_c, as follows:

- If $c = \bigvee_{i=1..k} l_i$, then $A_c = \{a_{l_i}\}_{i=1}^{k}$, with $pre(a_{l_i}) = \bigwedge_{j=1..k, j\neq i} \neg l_j$, and $\mathit{effects}(a_{l_i}) = l_i$.
- If $c = \oplus_{i=1..k} l_i$, then $A_c = \{a_{l_i}\}_{i=1}^{k}$, with $pre(a_{l_i}) = l_i$, and $\mathit{effects}(a_{l_i}) = \bigwedge_{j=1..k, j\neq i} \neg l_j$.

Joining Immediate Reasoning and Observations

In a PPOS we can split the propositions into 3 disjoint sets — propositions whose value is always known (e.g. the location of the agent in the Wumpus problem), propositions whose value may be unknown, but for which there is an observation action (e.g., the stench in cells near the Wumpus lair), and propositions whose value can not be observed (e.g., the location of the lair of the Wumpus). Note that we must have some way to reason about the values of the variables which can not be observed, as otherwise we can just ignore them.

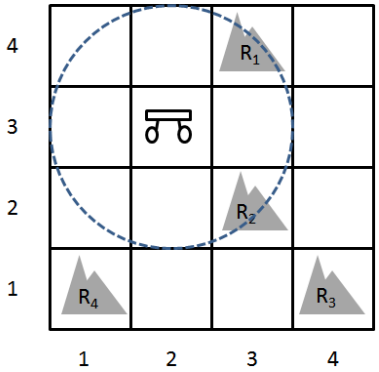

Figure 2: A Mars Rover rock sampling example, with a 4×4 grid and 4 rocks. The rover is at location $2, 3$, and the dotted line shows the range of its mineral sensor. Hence, only rocks 1 and 2 are within range at this location.

In many cases it is natural to define the value of the observable propositions through conditional effects over unobservable propositions. Consider for example the Mars Rover rock sample (Smith and Simmons, 2004; Brafman and Shani, 2012b), where the rover must sample rocks containing a desirable mineral in a grid (Figure 2). The rocks locations are known, but the agent must sense for the mineral using a long range sensor. The sensor reports the existence of the mineral in some rock within its range.

A natural formalization of this domain may have an action *activate-sensor-at-2-3* with preconditions *at-2-3* and conditional effects $\mathit{good\text{-}rock}_1 \rightarrow$ *good-rocks-in-range* and $\mathit{good\text{-}rock}_2 \rightarrow$ *good-rocks-in-range*, and an additional action *observe-rocks-in-range* which observes the *good-rocks-in-range* proposition which signifies the sensor's output. While these are separate actions, used together they allow us to reason about which rocks contain the good mineral. In this case, *at-x-y* is always known, *good-rocks-in-range* is unknown but directly observable, and $\mathit{good\text{-}rock}_i$ is unknown and not directly observable. We now explain how these two actions — *activate-sensor* and *observe-good-rocks-in-range* can be joined to allow us to observe and reason about certain propositions together.

Let a be an action that contains a set of conditional effects of the form (c_i, e) where c_i is unknown and unobservable and e is observable, and there is no other action that affects the value of e that is not mutually exclusive, i.e., that can be executed at the same state as a. In our example the *activate-sensor-at-x-y* actions are the only actions that affect *good-rocks-in-range*, and each such action requires the agent to be at a different location. For all such actions we consider the reverse of the conditions — $(e, \bigvee_i c_i)$.

We now create a new action $a \circ a_{obs}$ where a_{obs} is an observation action over e. The preconditions of the new action will be the conjunction of the preconditions of a and a_{obs}. The observation of $a \circ a_{obs}$ is e, and effects $\mathit{effects}(a \circ a_{obs}) = \mathit{effects}_u(a) \wedge \bigvee_i c_i$, where $\mathit{effects}_u(a)$ are the unconditional effects of a. When there is more than a single c_i, we remove the non-determinism, by creating a set of actions $a_i \circ a_{obs}$ whose preconditions $\mathit{pre}(a \circ a_{obs}) \wedge \bigwedge_{j \neq i} \neg c_j$, and whose effect is $\mathit{effects}(a_i \circ a_{obs}) = \mathit{effects}_u(a) \wedge c_i$. In the example above we have two such joined actions with preconditions *at-2-3* $\wedge \neg \mathit{good\text{-}rock}_j$, observation *good-rocks-in-range* and effect $\mathit{good\text{-}rock}_i$ where $i = 1$ and $j = 2$ or vice versa.

Note that this method is not general, in that there might be a sequence of unobservable propositions $p_1, ..., p_k$ and a set of actions a_i with conditional effects (p_i, p_{i+1}), and an observation action only for p_k. Our translation only allows reasoning over the value of p_{k-1}, and not about any of its predecessors. That being said, the only benchmark that exhibits this behavior is *localize*, while many other (e.g. *medpks, rock-sample*) contain the behavior we exploit above.

Finding Fact Landmarks

We now run a landmark detection algorithm on our relaxed domain that contains the actions created above, as well as all the "regular" actions that have no conditional effects, and operates only on the known propositions. In our experiments we used back-chaining using the possible first achievers (Richter, 2010). However, one can use any landmark detection algorithm on the relaxed problem described above. The only change in the landmark detection algorithms is with respect to the sensing actions, where we assume that they optimistically provide any required value of the observed proposition.

The Heuristic Contingent Planner Algorithm

We can now present our online contingent planning solver. The planner progresses by repeatedly identifying a reachable observation action that heuristically provides valuable information towards the achievement of the goal. The planner then plans in a classical setting to execute the observation action, assuming all unknown propositions to have a negative value. The plan is executed, followed by the observation action. Now, the process repeats with the additional information that was provided by the observation action. This process is repeated until the goal can be achieved without executing any additional observation actions. Algorithm 1 shows this high level algorithm.

Algorithm 2 shows the process for selecting the next sensing action. It estimates the myopic value of information of the observation action, i.e., how much value will be achieved from executing the action, ignoring future observations.

We first compute the set of achievable literals in the relaxed problem. Then, we see which observation actions can

Algorithm 1 Heuristic Contingent Planner

Input: π — a PPOS problem
1: $\pi_{relaxed} \leftarrow$ The relaxation of π as explained above
2: $L \leftarrow$ The set of landmarks for $\pi_{relaxed}$
3: $s \leftarrow$ All known literals at the initial state
4: **while** There is no classical plan that achieves G from s with no observation actions **do**
5: $a_{obs} \leftarrow ChooseNextSensingAction(\pi_{relaxed}, L, s)$
6: $P_{classic} \leftarrow Plan(s, pre(a_{obs}))$
7: $s' \leftarrow Execute(P_{classic}, s)$
8: $Execute(a_{obs})$ and observe literal l
9: $s \leftarrow s' \cup \{l\}$
10: **end while**
11: $P_{classic} \leftarrow Plan(s, G)$
12: $Execute(P_{classic}, s)$

be executed that sense the value of some unknown proposition. These are the candidate actions to be returned by the algorithm. To choose the heuristically best observation action, we analyze the value of observing p, by assuming that we have observed p, and computing which literals now become reachable. Then, we assume that we have observed $\neg p$ and compute again which literals become available.

Our policy for returning the heuristically best action first looks at the number of satisfied landmarks following the observation. Given multiple observation actions that satisfy the same number of landmarks, we break ties by looking at the sum of the number of literals and new observation actions that become achievable following the execution of the observation action. Finally, we break ties again in favor of the action which requires the minimal number of actions (in the relaxed domain) to execute.

Algorithm 2 Choosing the Next Sensing Action

Input: $\pi_{relaxed}$ — a relaxed PPOS problem, L a set of landmarks, s the set of currently known literals
1: $s' \leftarrow$ the set of achievable literals given $\pi_{relaxed}$ and s
2: $\Omega \leftarrow \{a : a \in A, pre(a) \in s', obs(a) \neq \phi, obs(a) \notin s'\}$
3: **for each** action $a \in \Omega$ **do**
4: $p \leftarrow obs(a)$, $s'_+ \leftarrow s' \cup \{p\}$, $s'_- \leftarrow s' \cup \{\neg p\}$
5: $s''_+ \leftarrow$ the set of achievable literals given $\pi_{relaxed}$ and s'_+
6: $s''_- \leftarrow$ the set of achievable literals given $\pi_{relaxed}$ and s'_-
7: $score_{landmarks}(a) \leftarrow$ the number of landmarks satisfied in s''_+ and s''_-, but not in s'
8: $score_{literals}(a) \leftarrow$ the number of literals achievable in s''_+ and s''_-, but not in s'
9: $score_{obs}(a) \leftarrow$ the number of sensing actions achievable in s''_+ and s''_-, but not in s'
10: $score_{cost}(a) \leftarrow$ the number of actions required from s before a can be executed in $\pi_{relaxed}$
11: $score(a) \leftarrow \langle score_{landmarks}(a), score_{literals}(a) + score_{obs}(a), score_{cost}(a) \rangle$
12: **end for**
13: **return** $argmax_{a \in \Omega} score(a)$

Empirical Evaluation

We now compare HCP to state-of-the-art online contingent planners, CLG (Albore, Palacios, and Geffner, 2009), SDR (Brafman and Shani, 2012b), MPSR (Brafman and Shani, 2012a), and K-Planner (Bonet and Geffner, 2011b) on various benchmarks. The experiments were conducted on a Windows Server 2008 machine with 24 2.66GHz cores (although each experiment uses only a single core) and 32GB of RAM. The underlying classical planner is FF (Hoffmann and Nebel, 2001).

Table 1 shows that HCP is much faster than all other planners, except for the K-Planner. The plan quality (number of actions) of HCP is also typically quite good. The only domain that could not be solved by HCP is *localize*, because it does not conform to our assumptions concerning reasoning about the hidden propositions.

K-Planner is very fast, but it can only run on domains where the hidden propositions remain constant throughout the execution. Thus, it is unsuitable for solving many of the benchmarks. Except for K-Planner, HCP is by far the fastest contingent planner, and in many cases improves runtime by more than an order of magnitude.

Related Work

Bonet and Geffner (2000) first proposed using heuristic search in belief space. Since then, several heuristics for belief states were proposed. Bryce, Kambhampati, and Smith (2006) argue that belief state heuristics typically aggregate distance estimates from the individual states which the belief state describes to the goal. Taking the maximum distance corresponds to assuming positive interaction (that is, the plan for reaching the goal from s1 also helps to reach the goal from s2), while summarizing the distances corresponds to assuming independence between these plans (that is, the plans for s1 and s2 neither help or interfere with each other).

Contingent-FF (CFF) (Hoffmann and Brafman, 2005) uses delete-relaxation, assuming *generous execution semantics*, which ignores actions whose preconditions are not satisfied during execution. They construct a relaxed conformant plan by building a variant of the relaxed planning graph, accounting for which facts are known at each layer. CFF represents and reasons about knowledge through a logic formula over the history. CFF also identifies *observation goals* — observations needed by a later action.

The DNF planner (To, Pontelli, and Son, 2009) uses a heuristic based on the number of satisfied goals, the cardinality of the belief state, and a measure called the *square distance* of the belief state to the goal, which is the sum of the number of unsatisfied goals in each individual state in the belief state. Goals are trivially also landmarks, and thus the number of unsatisfied goals can be seen as a special case of the number of unsatisfied landmarks, with a trivial landmark discovery method.

HCP is related to these planners as it also searches in belief space, although it doesn't explicitly represent and reasons about it. On the other hand, our use of landmark detection is far beyond any heuristics currently applied by other belief search planners.

Surprisingly, online planners that plan repeatedly once new knowledge has been acquired are less popular in this line of research. This reduces the scalability of these methods, because the complete plan tree can be exponential in the

Table 1: Comparing the performance of state of the art contingent planners. Blank cells represent problems that the planners were unable to solve. CSU denotes models that CLG can solve but cannot simulate execution for.

	HCP		MPSR		SDR		CLG		K-Planner	
Name	Actions	Time	Actions	Time	Actions	Time	Actions	Time	Actions	Time
cloghuge	55.48 (0.304)	**5.9** (0.0452)			61.17 (0.44)	117.13 (4.19)	**51.76** (0.33)	8.25 (0.08)		
ebtcs-70	42.32 (0.6712)	**1.12** (0.0188)	44.5 (0.7)	22.4 (0.3)	**35.52** (0.75)	3.18 (0.07)	**36.52** (0.86)	73.96 (0.14)		
elog7	**20** (0.076)	**0.32** (0.0016)	23.5 (0.1)	1.4 (0.1)	21.76 (0.07)	0.85 (0.01)	**20.12** (0.05)	1.4 (0.08)		
CB-9-5	**324** (2.24)	158.9 (1.76)			392.16 (2.81)	505.48 (8.82)	CSU		358.08 (15.8)	**94.18** (3.31)
CB-9-7	**425** (2.2636)	373 (2.28)			487.04 (2.95)	833.52 (15.82)	CSU		458.36 (14.64)	**116.63** (3.24)
doors13	**96.68** (0.52)	**30** (0.1296)	197.92 (1.2)	105.5 (2.1)	120.8 (0.93)	158.54 (2.01)	105.48 (0.89)	330.73 (0.21)	109.72 (4.76)	37.96 (1.72)
doors15	**137.9** (1.1052)	**52.6** (0.6228)	262.2 (1.9)	190 (3.3)	143.24 (1.36)	268.16 (3.78)			150.88 (4.7)	**55.24** (2)
doors17	**170** (1.456)	91 (0.708)	368.25 (3.4)	335.3 (5.3)	188 (1.64)	416.88 (6.16)			188.8 (5.79)	**79.24** (2.62)
localize17			59.8 (0.9)	**230.4** (7.7)	**45** (0.86)	928.56 (33.2)	CSU			
unix3	**40.48** (1.156)	**1.77** (0.0448)	69.7 (1.7)	5.2 (0.1)	56.32 (1.72)	5.47 (0.18)	51.32 (0.97)	18.56 (0.05)	**45.48** (4.59)	16.87 (1.56)
unix4	94.56 (1.88)	**20.21** (0.2868)	158.6 (4.3)	30.4 (1.1)	151.72 (4.12)	35.22 (0.94)	**90.8** (2.12)	189.41 (0.6)	87.04 (8.54)	38.81 (3.53)
Wumpus15	**65.08** (1.1052)	9.57 (0.11248)	**65** (1.6)	126.6 (3.1)	120.14 (2.4)	324.32 (7.14)	101.12 (0.67)	330.54 (0.25)	107.64 (4.6)	**7.17** (0.6)
Wumpus20	90 (1.3984)	34 (0.3396)	**71.6** (1.2)	261.1 (7)	173.21 (3.4)	773.01 (20.78)	155.32 (0.95)	1432 (0.47)	151.52 (6.29)	**16.03** (1)
RockSample 8-12	**105.76** (0.3984)	**6.3** (0.0496)			127.24 (0.68)	113.4 (0.79)				
RockSample 8-14	**135** (0.52)	**9** (0.038)			142.08 (0.8)	146.75 (1.19)				

worst case in the number of propositions. Thus, even fully specifying the plan tree may be impossible.

A second popular approach to contingent planning is the compilation-based approach (Albore, Palacios, and Geffner, 2009; Shani and Brafman, 2011; Brafman and Shani, 2012a; Bonet and Geffner, 2011a). These methods translate a PPOS into a classical planning problem, directly reasoning about the possible hidden state. Such methods add new "knowledge" propositions and modify actions so that the state space is transformed into a belief space, essentially allowing a classical planner to plan in belief space. This reduction allows leveraging advances in classical planning, such as recent, powerful heuristic generation methods. In this setting online approaches were developed, that plan only for branches of the plan tree that can be reached given the true hidden state at runtime.

Some of these methods make simplifying assumptions concerning the problem structure, as we do. For example CLG (Albore, Palacios, and Geffner, 2009) assumes limited uncertainty, modeled formally through the notion of *conformant-width*. Specifically, CLG cannot solve problems of width larger than 1, like Rock Sample. K-Planner Bonet and Geffner (2011a) has an even more strict assumption — it can solve only problems where the unknown proposition remain constant through the plan execution. This makes reasoning about the hidden state much easier, and our reasoning actions are inspired by this observation.

Conclusion and Future Work

We introduced a new approach to contingent planning, relying on heuristics computed over a relaxation of the domain description, without maintaining a belief state explicitly, or translating the problem into classical planning, which are the two popular approaches to contingent planning under partial observability.

Our planner, HCP, leverages certain properties of many benchmarks in order to avoid the explicit maintenance of belief states. Domains which do not conform to these properties, cannot be solved by HCP.

In the future we intend to generalize our assumptions concerning these properties, and formally identify domains where HCP works well, and provide a proof for its soundness and completeness for these domains. We will also look into less strict assumptions that may help us to solve domains which are currently impossible for HCP.

References

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

Blum, A., and Furst, M. L. 1997. Fast planning through planning graph analysis. *Artif. Intell.* 90(1-2):281–300.

Bonet, B., and Geffner, H. 2000. Planning with incomplete information as heuristic search in belief space. In *Proc. AIPS'00*, 52–61.

Bonet, B., and Geffner, H. 2011a. Planning under partial observability by classical replanning: Theory and experiments. In *IJCAI'11*.

Bonet, B., and Geffner, H. 2011b. Planning under partial observability by classical replanning: Theory and experiments. In *IJCAI*, 1936–1941.

Brafman, R. I., and Shani, G. 2012a. A multi-path compilation approach to contingent planning. In *AAAI*.

Brafman, R. I., and Shani, G. 2012b. Replanning in domains with partial information and sensing actions. *Journal of Artificial Intelligence Research (JAIR)* 45:565–600.

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

Hoffmann, J., and Brafman, R. I. 2005. Contingent planning via heuristic forward search with implicit belief states. In *ICAPS*, 71–80.

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

Hoffmann, J.; Porteous, J.; and Sebastia, L. 2004. Ordered landmarks in planning. *Journal of AI Research* 22:215–278.

Karpas, E., and Domshlak, C. 2009. Cost-optimal planning with landmarks. In *IJCAI*.

Keyder, E.; Richter, S.; and Helmert, M. 2010. Sound and complete landmarks for and/or graphs. In *Proceedings of the European Conference on Artificial Intelligence (ECAI)*, 335–340.

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

Richter, S., and Westphal, M. 2010. The LAMA planner: Guiding cost-based anytime planning with landmarks. *JAIR* 39:127–177.

Richter, S. 2010. *Landmark-Based Heuristics and Search Control for Automated Planning*. Ph.D. Dissertation, Griffith University.

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

Smith, T., and Simmons, R. 2004. Heuristic search value iteration for POMDPs. In *UAI 2004*.

To, S. T.; Pontelli, E.; and Son, T. C. 2009. A conformant planner with explicit disjunctive representation of belief states. In *ICAPS*.

To, S. T.; Pontelli, E.; and Son, T. C. 2011. On the effectiveness of cnf and dnf representations in contingent planning. In *IJCAI*, 2033–2038.

To, S. T.; Son, T. C.; and Pontelli, E. 2011. Contingent planning as and/or forward search with disjunctive representation. In *ICAPS*.

Zhu, L., and Givan, R. 2003. Landmark extraction via planning graph propagation. In *ICAPS 2003 Doctoral Consortium*, 156–160.