Conditional Planning in the Discrete Belief Space

Jussi Rintanen*
Albert-Ludwigs-Universität Freiburg, Institut für Informatik
Georges-Köhler-Allee, 79110 Freiburg im Breisgau
Germany

Abstract

Probabilistic planning with observability restrictions, as formalized for example as partially observable Markov decision processes (POMDP), has a wide range of applications, but it is computationally extremely difficult. For POMDPs, the most general decision problems about existence of policies satisfying certain properties are undecidable.

We consider a computationally easier form of planning that ignores exact probabilities, and give an algorithm for a class of planning problems with partial observability. We show that the basic backup step in the algorithm is NP-complete. Then we proceed to give an algorithm for the backup step, and demonstrate how it can be used as a basis of an efficient algorithm for constructing plans.

1 Introduction

When the sequence of states that will be visited during plan execution cannot be exactly predicted, for example because of nondeterminism, it is necessary to produce plans that apply different actions depending on how the plan execution has proceeded so far. Such plans are called conditional plans.

Construction of conditional plans is particularly difficult when there is no full observability; that is, when during plan execution it is not possible to uniquely determine what the current state of the world is. Planning problems having this property are said to be partially observable, and their solution requires that the sets of possible current world states – the belief states – are (implicitly) maintained during plan execution and (implicitly) represented by a plan.

The earliest work on planning with partial observability was in the framework of partially observable Markov decision processes (POMDPs) [Smallwood and Sondik, 1973; Kaelbling *et al.*, 1998]. Planning with POMDPs is computationally difficult. For unbounded horizon lengths an unbounded number of probability distributions corresponding to belief states needs to be considered, and finding optimal plans is not in general solvable [Madani *et al.*, 1999]. A natural approach for easing the computational difficulty of POMDP planning is to consider horizons of a bounded length [Mundhenk *et al.*, 2000]. A second approach which has been pursued with algorithms for conditional planning [Weld *et al.*, 1998; Bonet and Geffner, 2000; Bertoli *et al.*, 2001], ignores probabilities and hence directly yields a finitary problem. Main decision problems related to non-probabilistic planning with partial observability are 2-EXP-complete [Rintanen, 2004a].

A main difference between POMDPs and corresponding non-probabilistic problems is that the latter do not use probabilistic notions like success probability or expected cost, and require that a plan must reach the goals with certainty. An implication of success probability 1 is that uncertainty about observations and sensing can be ignored: if an observation is correct with a probability strictly less than 1 then it is as good as no observation at all.

For this planning problem we present an iterative algorithm that has some resemblance to iterative algorithms for solving POMDPs. The algorithm maintains a data structure representing those belief states for which a conditional plan has been shown to exist. Initially this data structure represents those belief states consisting of goal states only. Then this data structure is repeatedly extended by performing search backwards from the goal belief states.

The structure of the paper is as follows. Section 2 defines the planning problem. Sections 3 and 4 respectively describe the formal framework and analyze its properties. Section 5 proposes a planning algorithm and Section 6 presents experimental results obtained with an implementation of the algorithm. Section 7 concludes the paper.

2 The Planning Problem

In this section we present a formalization of planning in which states are atomic objects without internal structure.

Definition 1 *A* problem instance *is* $\langle S, I, O, G, P\rangle$ *where* S *is the set of states,* $I \subseteq S$ *is the set of initial states,* O *is the set of actions* $o \subseteq S \times S$*,* $G \subseteq S$ *is the set of goal states and* $P = (C_1, \ldots, C_n)$ *is a partition of* S *into classes of observationally indistinguishable states satisfying* $\bigcup\{C_1, \ldots, C_n\} = S$ *and* $C_i \cap C_j = \emptyset$ *for all* i, j *such that* $1 \leq i < j \leq n$*.*

Making an observation tells which set C_i the current state belongs to. Distinguishing states in a given C_i is not possible.

*This research was partly supported by DFG grant RI 1177/2-1.

An action is a relation between states and their successor states. An action o is *applicable* in a state s if sos' for some $s' \in S$. Define the *image* of a set B of states with respect to an action o as $\text{img}_o(B) = \{s' \in S | s \in B, sos'\}$. The *preimage* is $\text{preimg}_o(B) = \{s \in S | \emptyset \neq \text{img}_o(\{s\}) \subseteq B\}$, consisting of those states from which o is guaranteed to reach a state in B. An action o is deterministic if it is a partial function.

Plans are directed graphs with two kinds of nodes: action nodes and observation nodes.

Definition 2 *Let $\langle S, I, O, G, (C_1, \ldots, C_n)\rangle$ be a problem instance. A plan is a triple $\langle N, b, l\rangle$ where*

- *N is a finite set of nodes,*
- *$b \in N$ is the initial node,*
- *$l : N \to (O \times N) \cup 2^{2^S \times N}$ is a function that assigns each node an action and a successor node $\langle o, n\rangle \in O \times N$ or a set of states and successor nodes $\langle C, n\rangle \in 2^{2^S \times N}$ where $C = \bigcup\{C'_1, \ldots, C'_m\}$ for some $\{C'_1, \ldots, C'_m\} \subseteq \{C_1, \ldots, C_n\}$. In the first case the node is an action node and in the second an observation node.*

  For all $n \in N$ and $\{\langle C, m\rangle, \langle C', m'\rangle\} \subseteq l(n)$ the observations C and C' may not intersect: $C \cap C' = \emptyset$.

  Nodes with $l(n) = \emptyset$ are terminal.

We restrict to acyclic plans. Acyclicity means that the graph $\langle N, E\rangle$, where $\langle n, n'\rangle \in E$ iff $l(n) = \langle o, n'\rangle$ for some o or $\langle C, n'\rangle \in l(n)$ for some C, is acyclic.[1]

Plan execution starts from the initial node b and any of the initial states. For an action node with label $\langle o, n\rangle$ in state s execute o and continue from n and a state in $\text{img}_o(s)$. For an observation node identify $\langle C, n\rangle$ in the node label so that $s \in C$, and then continue from n and s. A plan solves a problem instance if all of its executions terminate in a terminal node and a goal state. Execution of an acyclic plan can have at most as many steps as there are nodes in the plan.

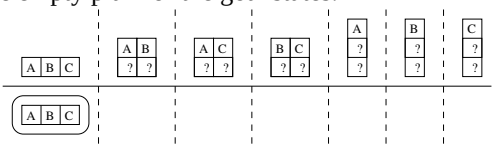
3 Problem Representation

Now we introduce the representation for sets of state sets for which a plan for reaching goal states exists.

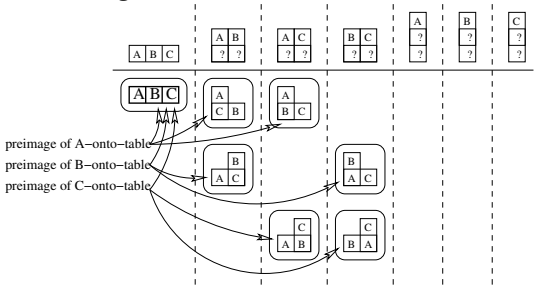
In the following example states are viewed as valuations of state variables, and the observational classes correspond to valuations of those state variables that are observable.

Example 3 Consider the blocks world with the state variables *clear(X)* observable, allowing to observe the topmost block of each stack. With three blocks there are 7 observational classes because there are 7 valuations of $\{\text{clear(A)}, \text{clear(B)}, \text{clear(C)}\}$ with at least one block clear.

Consider the problem of trying to reach the state in which all blocks are on the table. For each block there is an action for moving it onto the table from wherever it was before. If a block cannot be moved nothing happens. Initially we only have the empty plan for the goal states.

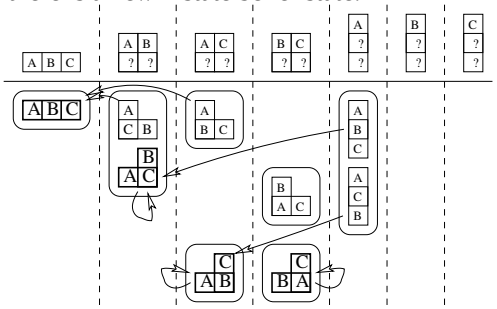


Then we compute the preimages of this set with actions that respectively put the blocks A, B and C onto the table, and split the resulting sets to the different observational classes.



Now for these 7 belief states we have a plan consisting of one or zero actions. But we also have plans for sets of states that are only represented implicitly. These involve branching. For example, we have a plan for the state set consisting of the four states in which respectively all blocks are on the table, A is on C, A is on B, and B is on A. This plan first makes observations and branches, and then executes the plan associated with the belief state obtained in each case. Because 3 observational classes each have 2 belief states, there are 2^3 maximal state sets with a branching plan. From each class only one belief state can be chosen because observations cannot distinguish between belief states in the same class.

We can find more belief states that have plans by computing preimages of existing belief states. Let us choose the belief states in which respectively all blocks are on the table, B is on C, C is on B, and C is on A, and compute their union's preimage with A-onto-table. The preimage intersected with the observational classes yields new belief states: for the class with A and B clear there is a new 2-state belief state covering both previous belief states in the class, and for the class with A clear there is a new 2-state belief state.



Computation of further preimages yields for each observational class a belief state covering all the states in that class, and hence a plan for every belief state. ■

The above example shows how the exponential number of state sets (corresponding to the Cartesian product of the observational classes) considered by Rintanen [2002] is represented only implicitly. The algorithm by Rintanen [2002] ex-

[1] Construction of cyclic plans requires looking at more global properties of transition graphs than what is needed for acyclic plans. The difficulties of cyclic plans in our framework are similar to those in the MDP/POMDP framework when using average rewards instead of finite horizons or discounted rewards [Puterman, 1994].

plicitly generates the state sets, the number of which in many cases is very high. With the new representation the computational complexity is shifted from the size of the representation to the time it takes to find a combination of belief states having a useful preimage. This shift is useful for two reasons. First, much of the space complexity (and the time complexity it implies) is traded to time complexity only: the state sets are not represented explicitly (except in the unobservable special case.) Second, the succinct representation allows much better control on which belief states to produce, and although finding one new belief state and plan still takes worst-case exponential time, this may be performed by clever algorithms and be further sped up by heuristics.

Next we formalize the framework in detail.

Definition 4 (Belief space) *Let $P = (C_1, \ldots, C_n)$ be a partition of the set of all states. Then* a belief space *is an n-tuple $\langle G_1, \ldots, G_n \rangle$ where $G_i \subseteq 2^{C_i}$ for all $i \in \{1, \ldots, n\}$ and $B \not\subset B'$ for all $i \in \{1, \ldots, n\}$ and $\{B, B'\} \subseteq G_i$.*

Notice that in each component of a belief space we only have set-inclusion maximal belief states. The simplest belief spaces are obtained from sets B of states as $\mathcal{B}(B) = \langle \{C_1 \cap B\}, \ldots, \{C_n \cap B\} \rangle$. A belief space is extended as follows.

Definition 5 (Extension) *Let $P = (C_1, \ldots, C_n)$ be the partition of all states, $G = \langle G_1, \ldots, G_n \rangle$ a belief space, and T a set of states. Define $G \oplus T$ as $\langle G_1 \Cup (T \cap C_1), \ldots, G_n \Cup (T \cap C_n) \rangle$ where the operation $\Cup$ adds the latter set of states to the former set of sets of states and eliminates sets that are not set-inclusion maximal, defined as $U \Cup V = \{R \in U \cup \{V\} | R \not\subset K$ for all $K \in U \cup \{V\}\}$.*

A belief space $G = \langle G_1, \ldots, G_n \rangle$ represents the set of sets of states $\text{flat}(G) = \{B_1 \cup \cdots \cup B_n | B_i \in G_i$ for all $i \in \{1, \ldots, n\}\}$ and its cardinality is $|G_1| \cdot |G_2| \cdot \ldots \cdot |G_n|$.

4 Complexity of Basic Operations

The basic operations on belief spaces needed in planning algorithms are testing the membership of a set of states in a belief space, and finding a set of states whose preimage with respect to an action is not contained in the belief space. Next we analyze the complexity of these operations.

Theorem 6 *For belief spaces G and state sets B, testing whether there is $B' \in$ flat(G) such that $B \subseteq B'$, and computing $G \oplus B$ takes polynomial time.*

Proof: Idea: A linear number of set-inclusion tests suffices. □

Our algorithm for extending belief spaces by computing the preimage of a set of states (Lemma 8) uses exhaustive search and runs in worst-case exponential time. This asymptotic worst-case complexity is very likely the best possible because the problem is NP-hard. Our proof for this fact is a reduction from SAT: represent each clause as the set of literals that are not in it, and then a satisfying assignment is a set of literals that is not included in any of the sets, corresponding to the same question about belief spaces.

Theorem 7 *Testing if for belief space G there is $R \in$ flat(G) such that preimg$_o(R) \not\subseteq R'$ for all $R' \in$ flat(G) is NP-complete. This holds even for deterministic actions o.*

Proof: Membership is easy: For $G = \langle G_1, \ldots, G_n \rangle$ choose nondeterministically $R_i \in G_i$ for every $i \in \{1, \ldots, n\}$, compute $R = \text{preimg}_o(R_1 \cup \cdots \cup R_n)$, and verify that $R \cap C_i \not\subseteq B$ for some $i \in \{1, \ldots, n\}$ and all $B \in G_i$. Each of these steps takes only polynomial time.

Let $T = \{c_1, \ldots, c_m\}$ be a set of clauses over propositions $A = \{a_1, \ldots, a_k\}$. We define a belief space based on states $\{a_1, \ldots, a_k, \hat{a}_1, \ldots, \hat{a}_k, z_1, \ldots, z_k, \hat{z}_1, \ldots, \hat{z}_k\}$. The states $\hat{a}$ represent negative literals. Define

$$\begin{array}{rcl} c'_i & = & (A \backslash c_i) \cup \{\hat{a} | a \in A, \neg a \notin c_i\} \text{ for } i \in \{1, \ldots, m\}, \\ G & = & \langle \{c'_1, \ldots, c'_m\}, \{\{z_1\}, \{\hat{z}_1\}\}, \ldots, \{\{z_k\}, \{\hat{z}_k\}\} \rangle, \\ o & = & \{\langle a_i, z_i \rangle | 1 \leq i \leq k\} \cup \{\langle \hat{a}_i, \hat{z}_i \rangle | 1 \leq i \leq k\}. \end{array}$$

We claim that T is satisfiable if and only if there is $B \in$ flat(G) such that preimg$_o(B) \not\subseteq B'$ for all $B' \in$ flat(G).

Assume T is satisfiable, that is, there is M such that $M \models T$. Define $M' = \{z_i | a_i \in A, M \models a_i\} \cup \{\hat{z}_i | a_i \in A, M \not\models a_i\}$. Now $M' \subseteq B$ for some $B \in$ flat(G) because from each class only one of $\{z_i\}$ or $\{\hat{z}_i\}$ is taken. Let $M'' = \text{preimg}_o(M') = \{a_i \in A | M \models a_i\} \cup \{\hat{a}_i | a_i \in A, M \not\models a_i\}$. We show that $M'' \not\subseteq B$ for all $B \in$ flat(G). Take any $i \in \{1, \ldots, m\}$. Because $M \models c_i$, there is $a_j \in c_i \cap A$ such that $M \models a_j$ (or $\neg a_j \in c_i$, for which the proof goes similarly.) Now $z_j \in M'$, and therefore $a_j \in M''$. Also, $a_j \notin c'_j$. As there is such an a_j (or $\neg a_j$) for every $i \in \{1, \ldots, m\}$, M'' is not a subset of any c'_i, and hence $M'' \not\subseteq B$ for all $B \in$ flat(G).

Assume there is $B \in$ flat(G) such that $D = \text{preimg}_o(B) \not\subseteq B'$ for all $B' \in$ flat(G). Now D is a subset of $A \cup \{\hat{a} | a \in A\}$ with at most one of a_i and $\hat{a}_i$ for any $i \in \{1, \ldots, k\}$. Define a model M such that for all $a \in A$, $M \models a$ if and only if $a \in D$. We show that $M \models T$. Take any $i \in \{1, \ldots, m\}$ (corresponding to a clause.) As $D \not\subseteq B$ for all $B \in$ flat(G), $D \not\subseteq c'_i$. Hence there is a_j or $\hat{a}_j$ in $D \backslash c'_i$. Consider the case with a_j ($\hat{a}_j$ goes similarly.) As $a_j \notin c'_i$, $a_j \in c_i$. By definition of M, $M \models a_j$ and hence $M \models c_i$. As this holds for all $i \in \{1, \ldots, m\}$, $M \models T$. □

5 Planning Algorithms

Based on the problem representation in the preceding section, we devise a planning algorithm that repeatedly identifies new belief states (and associated plans) until a plan covering the initial states is found. The algorithm in Figure 2 tests for plan existence; further book-keeping is needed for outputting a plan. The size of the plan is proportional to the number of iterations the algorithm performs, and outputting the plan takes polynomial time in the size of the plan. The algorithm uses the subprocedure *findnew* (Figure 1) for extending the belief space (this is the NP-hard subproblem from Theorem 7). Our implementation of the subprocedure orders sets $f_1, \ldots, f_m$ by cardinality in a decreasing order: bigger belief states are tried first. We also use a simple pruning technique for deterministic actions o: If preimg$_o(f_i) \subseteq$ preimg$_o(f_j)$ for some i and j such that $i > j$, then we may ignore f_i.

PROCEDURE findnew(o,A,F,H);
IF $F = \langle\rangle$ *AND* preimg$_o(A) \not\subseteq B$ for all $B \in$ flat(H)
THEN RETURN A;
IF $F = \langle\rangle$ *THEN RETURN* $\emptyset$;
F is $\langle\{f_1, \ldots, f_m\}, F_2, \ldots, F_k\rangle$ for some $k \geq 1$;
FOR i := 1 *TO* m *DO*
  B := findnew(o,$A \cup f_i$,$\langle F_2, \ldots, F_k\rangle$,$H$);
  IF $B \neq \emptyset$ *THEN RETURN* B;
END;
RETURN $\emptyset$

Figure 1: Algorithm for finding new belief states

PROCEDURE plan(I,O,G);
$H := \mathcal{B}(G)$;
progress := true;
WHILE progress and $I \not\subseteq I'$ for all $I' \in$ flat(H) *DO*
  progress := false;
  FOR EACH $o \in O$ *DO*
    B := findnew(o,$\emptyset$,H,H);
    IF $B \neq \emptyset$ *THEN*
      BEGIN
        $H := H \oplus$ preimg$_o(B)$;
        progress := true;
      END;
  END;
END;
IF $I \subseteq I'$ for some $I' \in$ flat(H) *THEN RETURN* true
ELSE RETURN false;

Figure 2: Algorithm for planning with partial observability

Lemma 8 *Let H be a belief space and o an action. The procedure call findnew(o,$\emptyset$,F, H) returns a set B' of states such that $B' =$ preimg$_o(B)$ for some $B \in$ flat(F) and $B' \not\subseteq B''$ for all $B'' \in$ flat(H), and if no such belief state exists it returns $\emptyset$.*

Proof: Sketch: The procedure goes through the elements $\langle B_1, \ldots, B_n\rangle$ of $F_1 \times \cdots \times F_n$ and tests whether preimg$_o(B_1 \cup \cdots \cup B_n)$ is in H. The sets $B_1 \cup \cdots \cup B_n$ are the elements of flat(F). The traversal through $F_1 \times \cdots \times F_n$ is by generating a search tree with elements of F_1 as children of the root node, elements of F_2 as children of every child of the root node, and so on, and testing whether the preimage is in H. The second parameter of the procedure represents the state set constructed so far from the belief space, the third parameter is the remaining belief space, and the last parameter is the belief space that is to be extended, that is, the new belief state may not belong to it. □

The correctness proof of the procedure *plan* consists of the following lemma and theorems. The first lemma simply says that extending a belief space H is monotonic in the sense that the members of flat(H) can only become bigger.

Lemma 9 *Assume T is any set of states and $B \in$ flat(H). Then there is $B' \in$ flat$(H \oplus T)$ so that $B \subseteq B'$.*

The second lemma says that if we have belief states in different observational classes such that each is included in a belief state of a belief space H, then there is a set in flat(H) that includes all these belief states.

Lemma 10 *Let $B_1, \ldots, B_n$ be sets of states so that for every $i \in \{1, \ldots, n\}$ there is $B_i' \in$ flat(H) such that $B_i \subseteq B_i'$, and there is no observational class C such that for some $\{i, j\} \subseteq \{1, \ldots, n\}$ both $i \neq j$ and $B_i \cap C \neq \emptyset$ and $B_j \cap C \neq \emptyset$. Then there is $B' \in$ flat(H) such that $B_1 \cup \cdots \cup B_n \subseteq B'$.*

The proof of the next theorem shows how the algorithm is capable of finding any plan by constructing it bottom up starting from the leaf nodes. The construction is based on first assigning a belief state to each node in the plan, and then showing that the algorithm reaches that belief state from the goal states by repeated computation of preimages.

Theorem 11 *Whenever there exists a finite acyclic plan for a problem instance, the algorithm in Figure 2 returns* true.

Proof: Assume there is a plan $\langle N, b, l\rangle$ for a problem instance $\langle S, I, O, G, P\rangle$. Label all nodes of the plan as follows. The root node b is labeled with I, that is, $Z(b) = I$. When all parent nodes of a node n have a label, we assign a label to n. Let $l(n_1) = \langle o_1, n\rangle, \ldots, l(n_m) = \langle o_m, n\rangle$ for action nodes $n_1, \ldots, n_m$ that have n as the child node, and let $l(n_1') = \{\langle C_1, n\rangle, \ldots\}, \ldots, l(n_k') = \{\langle C_k, n\rangle, \ldots\}$ for all observation nodes $n_1', \ldots, n_k'$ with n as one of the children. Then $Z(n) =$ img$_{o_1}(Z(n_1)) \cup \cdots \cup$ img$_{o_m}(Z(n_m)) \cup (Z(n_1') \cap C_1) \cup \cdots \cup (Z(n_k') \cap C_k)$. If the above labeling does not assign anything to a node n, then assign $Z(n) = \emptyset$. Each node is labeled with those states that are possible in that node on some execution.

We show that if plans for $Z(n_1), \ldots, Z(n_k)$ exist, where $n_1, \ldots, n_k$ are children of a node n in a possible plan, then the algorithm determines that a plan for $Z(n)$ exists as well.

Induction hypothesis: For each plan node n such that all paths to a terminal node have length i or less, its label $B = Z(n)$ is a subset of some $B' \in$ flat(H) where H is the value of the program variable H after the *while* loop exits and H could not be extended further.

Base case $i = 0$: Terminal nodes of the plan are labeled with subsets of G. By Lemma 9, there is G' such that $G \subseteq G'$ and $G' \in$ flat(H) because initially $H = \mathcal{B}(G)$ and thereafter it was repeatedly extended.

Inductive case $i \geq 1$: Let n be a plan node. By the induction hypothesis for all children n' of n, $Z(n') \subseteq B$ for some $B \in$ flat(H).

If n is an observation node with children $n_1, \ldots, n_k$ and respective observations $C_1, \ldots, C_k$, then $Z(n) \cap C_1$, …, $Z(n) \cap C_k$ all occupy disjoint observational classes and superset of $Z(n) \cap C_i$ for every $i \in \{1, \ldots, k\}$ is in flat(H). Hence by Lemma 10 $Z(n) \subseteq B$ for some $B \in$ flat(H).

If n is an action node with action o and child node n', then img$_o(Z(n)) \subseteq Z(n')$, and by the induction hypothesis $Z(n') \subseteq B'$ for some $B' \in$ flat(H). We have to show that $Z(n) \subseteq B''$ for some $B'' \in$ flat(H). Assume that there is no such B''. But now by Lemma 8 findnew(o,$\emptyset$,H,H) would return B''' such that preimg$_o(B''') \not\subseteq B$ for all $B \in$ flat(H),

and the *while* loop could not have exited with H, contrary to our assumption about H. □

Theorem 12 *Let* $\Pi = \langle S, I, O, G, P \rangle$ *be a problem instance. If plan(I,O,G) returns* true, *then* Π *has a solution plan.*

Proof: Let $H_0, H_1, \ldots$ be the sequence of belief spaces H produced by the algorithm. We show that for all $i \geq 0$, for every $B \in \text{flat}(H_i)$ there is a plan that reaches G.

Induction hypothesis: H_i contains only such state sets $B \in \text{flat}(H_i)$ for which a plan reaching G exists.

Base case $i = 0$: $H_0 = \mathcal{B}(G)$, and the only state set in H_0 is G. The empty plan reaches G from G.

Inductive case $i \geq 1$: H_{i+1} is obtained as $H_i \oplus \text{preimg}_o(B)$ where $B = \text{findnew}(o, \emptyset, H_i, H_i)$.

Because by Lemma 8 $B \in \text{flat}(H_i)$, by induction hypothesis there is a plan π for B. The plan that executes o followed by π reaches G from $\text{preimg}_o(B)$.

Let B be any member of $\text{flat}(H_{i+1})$. We show that for B there is a plan for reaching G. The plan for B starts by a branch[2]. We show that for every possible observation, corresponding to one observational class, there is a plan that reaches G. Let C_j be the jth observational class. When observing C_j, the current state is in $B_j = B \cap C_j$. Now for B_j there is $B'_j \in H_{i+1,j}$ with $B_j \subseteq B'_j$ where $H_{i+1,j}$ is the jth component of H_{i+1}. Now by induction hypothesis there is a plan for B'_j if $B'_j \in H_{i,j}$, and if $B'_j \in H_{i+1,j} \backslash H_{i,j}$, then for the branch corresponding to C_j we use the plan for $\text{preimg}_o(B)$, as B'_j must be $\text{preimg}_o(B) \cap C_j$. □

Until now we have used only one partition of the state space to observational classes. However, it is relatively straightforward to generalize the above definitions and algorithms to the case in which several partitions are used, each for a different set of actions. This means that the possible observations depend on the action that has last been taken.

6 Experimentation with an Implementation

We have implemented the algorithm from the previous section and call the resulting planning system BBSP. The only heuristic is the one described in the preceding sections: *findnew* chooses bigger belief states first. The implementation is based on representing sets of states and actions as BDDs [Burch *et al.*, 1994]. There is a small improvement in the belief space representation with BDDs: all components of a belief space consisting of one belief state only are represented by one BDD.

We carried out a comparison to the MBP planner [Bertoli *et al.*, 2001] which uses forward-search together with some heuristics for restricting branching. MBP starts search from the initial states, and proceeds forward by taking actions, leading to another set of states, or by using observations to split the current state set to several smaller ones. Different choices of actions and observations induce a search tree.

[2]Some branches might not be needed, and if the intersection of B with only one observational class is non-empty the plan could start with an action node instead of a degenerate observation node.

		runtime in seconds		iterations
problem	$\|S\|$	MBP	BBSP	BBSP
BTCS1601	98	**0.60**	1.21	32
BTCS1701	104	**0.79**	1.38	34
BTCS1801	110	**1.01**	2.08	36
BTCS1901	116	**1.22**	2.28	38
BTCS2001	122	**1.44**	2.52	40
medical18	148	7.34	**2.49**	21
medical20	164	24.13	**2.91**	23
medical22	180	60.53	**3.48**	25
medical24	196	$>$ 20 m	**4.11**	27
medical26	212	$>$ 20 m	**6.61**	29
emptyroom07	49	**0.09**	0.38	41
emptyroom08	64	**0.12**	0.63	53
emptyroom10	100	**0.16**	1.56	81
emptyroom15	225	**0.37**	9.60	198
emptyroom20	400	**0.62**	25.58	243
ring03	162	**0.09**	0.16	11
ring04	648	**0.38**	**0.44**	19
ring05	2430	1.99	**1.03**	23
ring06	8748	13.12	**1.63**	27
ring07	30618	94.73	**2.41**	31
ring08	104976	744.90	**3.34**	35
ring09	354294	$>$ 20 m	**5.55**	39
ring10	1180980	$>$ 20 m	**7.06**	43
BW03fo	13	32.64	**0.14**	9
BW04fo	73	$>$ 20 m	**0.67**	22
BW05fo	501	$>$ 20 m	**6.43**	46
BW06fo	4051	$>$ 20 m	**133.14**	64
BW03pfo	13	**0.13**	**0.12**	9
BW04pfo	73	90.15	**0.69**	22
BW05pfo	501	$>$ 20 m	**6.29**	46
BW06pfo	4051	$>$ 20 m	**133.88**	64
BW03po	13	**0.08**	**0.12**	9
BW04po	73	**0.71**	**0.64**	22
BW05po	501	13.49	**6.41**	46
BW06po	4051	394.21	**198.85**	64

Table 1: Runtime comparison BBSP vs. MBP

Some of the MBP runtimes given by Bertoli et al. [2001] are much better than given by us in this paper (specifically on the medical and ring problems) because the branching heuristic used by Bertoli et al. works well on their formulations of the benchmarks: branch only on one observable state variable if possible. We used a slightly different formulation where one many-valued observation is replaced by a small number of Boolean observations.

Runtimes of the planners are given in Table 1. The runs were on a 360 MHz Sun Sparcstation under Solaris. The problem instances are the same as in [Rintanen, 2002] where MBP was shown in almost all cases to be faster than GPT [Bonet and Geffner, 2000] and much faster than the YKÄ planner [Rintanen, 2002] on most of the problems except the blocks worlds problems with full or almost full observability. BTCS is the bomb-in-the-toilet problem with sensing. In the medical problems patients are cured by performing tests and medicating. The emptyroom problems are about navigating from an unknown location to the center of the room. The ring problems are about closing and locking all the windows of a building consisting of a cycle of rooms. BW is the blocks

world with increasing number of blocks with the goal to arrange them into one stack from any initial configuration under different degrees of observability: *fo* is full observability, *pfo* is with the *on* relation observable, and *po* is partial observability (*clear* and *ontable* are observable). The problems are solvable under partial observability because moving a block only requires that it is clear and a move action is applicable no matter where the block is.

The rightmost column gives the number of iterations BBSP needs for finding a plan. MBP runtimes in some cases grow faster because it performs more search. This is most obvious in some of the problems with more observations as the number of possible ways of branching becomes astronomical. In our algorithm, the dynamic programming character of plan generation better avoids this explosion in the number of belief states to be considered.

In forward search there is an inherent conflict between A) keeping the size of plans and search trees down by *not* branching on all possible observations and B) branching enough to be able to find a plan. In MBP a number of heuristics is used for controlling branching, and for these benchmarks the heuristics in most cases work very well. For backward plan construction no similar conflict exists, and plan construction by a form of dynamic programming leads to effective reuse of already constructed belief states and plans, and there is no separate problem of deciding how to branch.

7 Conclusions and Future Work

We have presented a novel framework for non-probabilistic conditional planning with partial observability, proposed a backward search algorithm for finding plans, and shown that the basic backup step of the algorithm is NP-hard. We have also demonstrated how an efficient implementation of the backup step leads to competitive planning.

For future work we propose considering the problem of finding plans with quality guarantees, for example, plans with optimal execution length, as well as planning under more general infinite-horizon executions. Also, the use of more sophisticated heuristics for driving the planning algorithm should be considered, for example the ones proposed by Rintanen [2004b] generalized to partially observable problems.

References

[Bertoli *et al.*, 2001] Piergiorgio Bertoli, Alessandro Cimatti, Marco Roveri, and Paolo Traverso. Planning in nondeterministic domains under partial observability via symbolic model checking. In Bernhard Nebel, editor, *Proceedings of the 17th International Joint Conference on Artificial Intelligence*, pages 473–478. Morgan Kaufmann Publishers, 2001.

[Bonet and Geffner, 2000] Blai Bonet and Héctor Geffner. Planning with incomplete information as heuristic search in belief space. In Steve Chien, Subbarao Kambhampati, and Craig A. Knoblock, editors, *Proceedings of the Fifth International Conference on Artificial Intelligence Planning Systems*, pages 52–61. AAAI Press, 2000.

[Burch *et al.*, 1994] J. R. Burch, E. M. Clarke, D. E. Long, K. L. MacMillan, and D. L. Dill. Symbolic model checking for sequential circuit verification. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 13(4):401–424, 1994.

[Kaelbling *et al.*, 1998] Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1-2):99–134, 1998.

[Madani *et al.*, 1999] Omid Madani, Steve Hanks, and Anne Condon. On the decidability of probabilistic planning and infinite-horizon partially observable Markov decision problems. In *Proceedings of the 16th National Conference on Artificial Intelligence (AAAI-99) and the 11th Conference on Innovative Applications of Artificial Intelligence (IAAI-99)*, pages 541–548. AAAI Press, 1999.

[Mundhenk *et al.*, 2000] Martin Mundhenk, Judy Goldsmith, Christopher Lusena, and Eric Allender. Complexity of finite-horizon Markov decision process problems. *Journal of the ACM*, 47(4):681–720, 2000.

[Puterman, 1994] M. L. Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 1994.

[Rintanen, 2002] Jussi Rintanen. Backward plan construction under partial observability. In Malik Ghallab, Joachim Hertzberg, and Paolo Traverso, editors, *Proceedings of the Sixth International Conference on Artificial Intelligence Planning Systems*, pages 173–182. AAAI Press, 2002.

[Rintanen, 2004a] Jussi Rintanen. Complexity of planning with partial observability. In Shlomo Zilberstein, Jana Koehler, and Sven Koenig, editors, *ICAPS 2004. Proceedings of the Fourteenth International Conference on Automated Planning and Scheduling*, pages 345–354. AAAI Press, 2004.

[Rintanen, 2004b] Jussi Rintanen. Distance estimates for planning in the discrete belief space. In *Proceedings of the 18th National Conference on Artificial Intelligence (AAAI-2004) and the 13th Conference on Innovative Applications of Artificial Intelligence (IAAI-2004)*, pages 525–530. AAAI Press, 2004.

[Smallwood and Sondik, 1973] Richard D. Smallwood and Edward J. Sondik. The optimal control of partially observable Markov processes over a finite horizon. *Operations Research*, 21:1071–1088, 1973.

[Weld *et al.*, 1998] Daniel S. Weld, Corin R. Anderson, and David E. Smith. Extending Graphplan to handle uncertainty and sensing actions. In *Proceedings of the 15th National Conference on Artificial Intelligence (AAAI-98) and the 10th Conference on Innovative Applications of Artificial Intelligence (IAAI-98)*, pages 897–904. AAAI Press, 1998.