

# Backward Plan Construction under Partial Observability

Jussi Rintanen

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

## Abstract

We present algorithms for partially observable planning that iteratively compute belief states with an increasing distance to the goal states. The algorithms handle nondeterministic operators, but restrict to problem instances with a finite upper bound on execution length, that is plans without loops. We discuss an implementation of the algorithms which uses binary decision diagrams for representing belief states. It turns out that generation of new belief states from existing ones, corresponding to the use of conditional branches in the plans, can very naturally be represented as standard operations on binary decision diagrams. We also give a preliminary experimental evaluation of the algorithms.

## Introduction

For solving planning problems in which the exact sequence of states encountered during plan execution cannot be 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 tricky 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.

In this paper we address the partially observable conditional planning problem. Partial observability is a notoriously difficult problem in AI planning and in policy construction for partially observable Markov decision processes (POMDPs) (Sondik 1978; Kaelbling, Littman, & Cassandra 1998). Recent works for planning not directly based on traditional techniques for solving POMDPs include (Weld, Anderson, & Smith 1998; Rintanen 1999; Bonet & Geffner 2000; Bertoli *et al.* 2001). Bonet and Geffner handle numeric probabilities, like the POMDP model, the other works essentially handle three qualitative probabilities,  $p = 1.0$ ,  $p = 0.0$  and  $1.0 > p > 0.0$ . There is also some

work on an easier special case, the unobservable (or conformant) planning problem, which does not allow observations at all (Smith & Weld 1998; Cimatti & Roveri 2000; Bonet & Geffner 2000).

We present two new algorithms for partially observable planning that use backward search in the belief space, producing belief states with an increasing distance to the goal states, each corresponding to a plan with which the goals can be reached from the belief state in question. The backward steps in the construction correspond to operator applications (the maximal possible predecessor belief state for reaching a given belief state with an operator) and branching (belief states obtained as combination of a number of belief states that can be observationally distinguished from each other.)

An effective implementation of the algorithms is obtained by using BDDs for representing the belief states. The size of a BDD often does not grow proportionally to the number of states in the belief state that is represented. Also, the backward steps corresponding to branching allow a very effective handling of large number of potential ways of branching as simple BDD operations, sometimes leading to big reductions in the amount of work needed.

In two special cases, planning with full observability and with no observability, the algorithms function like some existing specialized algorithms for the problems in question.

If the problem instance is fully observable, the algorithms essentially do breadth-first search backwards from the goal states with the whole search tree traversed up to a certain level represented as a single BDD, corresponding to one belief state. This backward traversal of the state space is like in BDD-based algorithms for model-checking in computer-aided verification and in earlier algorithms for fully observable planning that use BDDs.

If the problem instance is unobservable, the computational problem is to find a path from the initial belief state to the goal belief state, like in classical planning, but at the level of belief states instead of states. The symbolic breadth-first traversal of the state space used in the fully observable case is not possible, and the algorithms just produce belief states with an increasing distance to the goal states.

Both of these two forms of plan search are realized within the planning algorithms we propose. We generate belief states backwards starting from the goal belief state. The belief states can be combined (except in the unobservable spe-

cial case) to obtain bigger belief states (the behavior used in the fully observable case without restrictions), and new belief states can be obtained by computing the possible predecessor belief states of a belief state with respect to an operator application (which is the only behavior used in the unobservable special case.)

The structure of the paper is as follows. The next section describes the exact planning problem we are addressing, including the type of observations we handle and the form of plans. Then we describe the combination operator for belief states which is the basis of the backward traversal algorithms, and how binary decision diagrams lead to a natural and efficient implementation of the combination operation. Two BDD-based algorithms based on the combination operation are described, the first uses exhaustive generation of belief states and the second uses a simple heuristic for selecting which belief states to produce. In the implementation section we describe the main insights obtained from implementing the algorithms, and make runtime comparisons to other planners. Finally, we conclude the paper by discussing connections to earlier work.

## Problem Definition

### Observation Model

During plan execution, before deciding which action to take next, it is often useful or necessary to try to determine what is the current state. Of course, given that the last action was taken in a certain belief state, it is possible to compute the set of possible successor states, but in addition, it may be possible to obtain new information about the world that allows distinguishing between the possible new current states.

The observation model we use is based on partitioning the state space  $S$  into non-empty pairwise-disjoint observational classes  $P = \langle C_1, \dots, C_n \rangle$  such that  $S = \bigcup_{i \in \{1, \dots, n\}} C_i$ . Given the actual current state  $s$  (which we in general do not know), we observe  $C_i$  for  $i \in \{1, \dots, n\}$  such that  $s \in C_i$ , meaning that the actual current state is one of the states in  $C_i$ , but no further distinctions between the states in  $C_i$  can be made. If we already knew that the current state is one of the states in  $B$ , then we know that  $s$  is in  $B \cap C_i$ .

This can easily be generalized to the case in which the possible observations depend on the last action taken; that is, we have different partitions for different operators, allowing for example the expression of special observation actions.

We have problem representations with state variables and implementation techniques that use BDDs in mind, so it is most convenient to assume that partitions  $C_1, \dots, C_n$  are induced by sets  $O$  of state variables that are observable. Then, a component  $C_i$  of the partition is a set  $C_i \subseteq S$  of states that assign the same values to all of the variables in  $O$ . Notice that when there are  $n$  observable Boolean state variables, there can be  $2^n$  components in the partition.

There are several extensions of the observation model that can be implemented by reduction to the basic model we described above. For example compound observations can be reduced to atomic observations, assuming a sufficiently expressive language for describing operators. Within the BDD framework, handling observability of compound formulae

$\phi$  directly is easy by having a state variable  $o_\phi$  observable, and requiring that  $o_\phi \leftrightarrow \phi$  holds in every state. This can be directly encoded to the BDD representation of transition relations of operators.

### Plans

The plans our algorithms produce are directed acyclic graphs. Terminal nodes correspond to goal belief states. Nodes with exactly one successor node are labeled with an operator, and they correspond to operator applications. Nodes with more than one successor are branches: one of the successor nodes is chosen based on what is observed. The edges to the successor nodes are labeled with corresponding sets of observations.

The algorithms in this paper do not produce plans with loops. Loops are needed for finitely representing plans that have no finite upper bound on execution length. Execution length may be unbounded when some of the operators are nondeterministic. For example, the number of times a dice has to be thrown to get 6 is unbounded, and the plan needs a loop (throw the dice until you get 6.) Loops are not needed when all operators are deterministic. Many types of nondeterministic problems have solutions as plans without loops.

## Distance Computation for Partially Observable Planning

Many of the earlier algorithms for fully observable planning are best understood as computing the distance/cost from every state to a goal state. These distances can be understood as a plan. The plan can be executed by always choosing an operator that reduces the distance of the current state by 1, or in nondeterministic planning, that makes the highest expected reduction in the distance/cost to a goal state.

In this paper we generalize this idea to the more complicated partially observable case. The passage from fully observable to partially observable planning requires employing the notion of belief states, that is sets of states, and computing (an upper bound on) the distances between belief states and the goal belief state. As in the fully observable case, the distance computation yields a plan as a byproduct.

The first complication is that the number of belief states is much higher than the number of states: for  $n$  state variables the maximum number of reachable states is  $2^n$ , and this induces a belief space consisting of  $2^{2^n}$  belief states.

The second complication is the definition of distances. We define the distance between a belief state and the goal belief state in terms of a plan implicitly associated with the belief state. The plan determines the distance as the maximum number of actions the plan may need for reaching a goal state. The computation of distances takes place in parallel with the computation of the belief states, and the plans need not be explicitly constructed.

In fully observable planning with deterministic actions, the distance of a state is simply one plus the minimum of the distances of the states that can be reached by one action. The same holds in partially observable planning for belief states, but there is also the additional possibility of determining the distance of a belief states in terms of the distances of other

belief states. This is because plans can have branches. If from  $B$  we can reach belief states having distance  $d$  by a branch (observation) node in the plan, without taking any actions, then the distance of  $B$  is also  $d$ .

So for the distance computation we need to take into account the branch nodes, and this is a main difference between fully observable and partially observable planning. In fully observable planning, if we know that plans for reaching the goals exist for both  $B_1$  and  $B_2$ , then we immediately know that there is also a plan for  $B_1 \cup B_2$ . In partially observable planning this is not the case: the existence of two plans, one reaching the goals from  $B_1$  and the other from  $B_2$  does not mean that there is a plan for  $B_1 \cup B_2$ . This is because in  $B_1 \cup B_2$  we might not be able to choose the appropriate subplan corresponding to  $B_1$  and  $B_2$ , as we might not know whether the state we are in belongs to  $B_1$  or to  $B_2$ . In the next section we make this problem explicit, and explain how plan branching is handled in the distance computation.

For making the distance computation more feasible we will restrict to generation of set-inclusion maximal belief states. When we know that belief state  $B$  has distance  $n$ , we do not need to separately represent belief states  $B' \subset B$  with distance  $n$ . This is because the same plan that reaches the goals from  $B$  will reach the goals also from  $B'$ .

### Combination Operation $\otimes$ for Belief States

Belief states  $B_1$  and  $B_2$  are *observationally distinguishable* if every  $s_1 \in B_1$  and  $s_2 \in B_2$  assign a different truth-value to at least one observable state variable, and therefore belong to different components of the partition representing observability (for all  $C_i$  in the partition, either  $C_i \cap B_1 = \emptyset$  or  $C_i \cap B_2 = \emptyset$ .) This means that after making the possible observations concerning the current state, one can exclude one of the sets of states  $B_1$  and  $B_2$  as impossible.

Later in Definition 2 we introduce a combination operator for belief states that is based on the following intuitions. If plan existence for belief states  $B_1$  and  $B_2$  has been shown (that is, there is plan  $Z_1$  for  $B_1$  and plan  $Z_2$  for  $B_2$ , then for  $B'_1 \subseteq B_1$  and  $B'_2 \subseteq B_2$  such that  $B'_1$  and  $B'_2$  are observationally distinguishable, there must also be a plan for  $B' = B'_1 \cup B'_2$  that reaches the goal states. This means that in belief state  $B'$  we can choose one of  $Z_1$  and  $Z_2$  on the basis of values of the observable state variables, and have a guarantee that the goal states will be reached by the respective subplans. Hence a plan for  $B'$  starts with a branch node that selects one of the subplans  $Z_1$  and  $Z_2$ .

Identifying sets  $B'_1$  and  $B'_2$  is easy in two special cases. If nothing is observable (the partition has only one component consisting of all states), the only possibilities are  $B'_1 = B_1, B'_2 = \emptyset$  and  $B'_1 = \emptyset, B'_2 = B_2$ ; that is, combining belief states does not produce new ones. If everything is observable (the partition consists of singleton sets), we choose  $B'_1 = B_1, B'_2 = B_2$ , which means that belief states can always be combined by taking their union.

In these two special cases the number of new belief states (1 or 2) is much smaller than in the general partially observable case, which will be discussed next. This agrees with the computational complexity results that say that the general planning problem with partial observability is much more

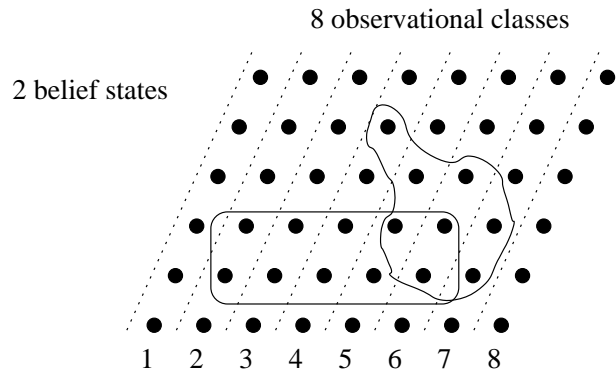


Figure 1: The intersections of two observational classes with the belief states are not in the inclusion relation. The belief states can be combined in four different ways.

difficult than either of these special cases, and that planning without observability is more difficult than planning with full observability (Mundhenk *et al.* 2000).

**Example 1** Consider the two belief states  $B_1$  and  $B_2$  in Figure 1. There are eight observational classes (components of the partition.) The problematic observational classes are those that intersect both  $B_1$  and  $B_2$  and the intersections are not in the inclusion relation (like in the unobservable case with only one observational class; in the fully observable case non-empty intersections coincide because they are singleton sets.) These are the classes  $C_4$  and  $C_5$ . For all other classes  $C_i, i \in \{1, 2, 3, 6, 7, 8\}$  either  $B_1 \cap C_i \subseteq B_2 \cap C_i$  or  $B_2 \cap C_i \subseteq B_1 \cap C_i$ .

A belief state containing both  $B_1 \cap C_j$  and  $B_2 \cap C_j$  for  $j \in \{4, 5\}$  is not (directly) backward reachable from  $B_1$  and  $B_2$ , because for none of the states in  $(B_1 \cap C_j) \cup (B_2 \cap C_j)$  can we say whether we are in  $B_1$  or  $B_2$ , and hence the appropriate plan associated with  $B_1$  or  $B_2$  cannot be chosen.

Therefore, any backward reachable belief state  $B$  must fulfill either  $B \cap C_j \subseteq B_1 \cap C_j$  or  $B \cap C_j \subseteq B_2 \cap C_j$ . Because we are interested in maximal belief states, this is either  $B \cap C_j = B_1 \cap C_j$  or  $B \cap C_j = B_2 \cap C_j$ .

Hence we reach backwards the following belief states.

$$\begin{aligned} B_{11} &= B_s \cup (B_1 \cap C_4) \cup (B_1 \cap C_5) \\ B_{12} &= B_s \cup (B_1 \cap C_4) \cup (B_2 \cap C_5) \\ B_{21} &= B_s \cup (B_2 \cap C_4) \cup (B_1 \cap C_5) \\ B_{22} &= B_s \cup (B_2 \cap C_4) \cup (B_2 \cap C_5) \end{aligned}$$

Here  $B_s = (B_1 \cup B_2) \cap (C_1 \cup C_2 \cup C_3 \cup C_6 \cup C_7 \cup C_8)$  is the union of the intersections of  $B_1$  and  $B_2$  with the unproblematic classes (intersections are in the inclusion relation). In these classes we can always safely choose the belief state with the bigger intersection.

Clearly, the number of combined belief states is  $2^n$  when the number of non-inclusion observational classes is  $n$ . ■

Notice that the number of observational classes may be exponential on the number of state variables, which may make the number of combinations extremely high.

Now we are ready to give the definition of the combination operator  $\otimes$  for belief states  $B_1$  and  $B_2$  which identifies maximal subsets of  $B_1 \cup B_2$  from which either  $B_1$  and  $B_2$  can be chosen based on the observations that can be made.

**Definition 2** Let  $P = \langle C_1, \dots, C_n \rangle$  be a partition of the set of all states. Let  $B_1$  and  $B_2$  be two sets of states. Then  $B_1 \otimes B_2$  is defined as

$$B_1 \otimes B_2 = \{I_1 \cup \dots \cup I_n \mid \begin{array}{l} I_i \in \{B_1 \cap C_i, B_2 \cap C_i\}, \\ I_i \not\subset B_1 \cap C_i, I_i \not\subset B_2 \cap C_i, \\ \text{for all } i \in \{1, \dots, n\} \}. \end{array}$$

The combination operator has many algebraic properties that may be taken advantage of in improving algorithms. In this section we point out the most important ones. We consider the operator  $\otimes$  with an arbitrary (but fixed) partition of the state space to observational classes.

**Theorem 3 (Monotonicity)** Let  $B$ ,  $B_1$  and  $B_2$  be belief states so that  $B_1 \subseteq B_2$ . For all  $B'_1 \in B \otimes B_1$ , there is  $B'_2 \in B \otimes B_2$  such that  $B'_1 \subseteq B'_2$ .

*Proof:* Let  $B'_1$  be any member of  $B \otimes B_1$ . Hence  $B'_1 = \bigcup_{1 \leq i \leq n} I_i$  for  $I_i$  such that  $I_i = B \cap C_i$  or  $I_i = B_1 \cap C_i$ .

Now we can construct  $B'_2$  as follows. The construction for every component in the partition (for every observational class) is independent of the other components. Choose any  $i \in \{1, \dots, n\}$ . If  $B'_1 \cap C_i = B \cap C_i$  and  $B \cap C_i \not\subset B_2 \cap C_i$ , we choose  $B \cap C_i$  for  $B'_2$ . Otherwise we choose  $B_2 \cap C_i$  for  $B'_2$ . In both cases the intersection  $B'_1 \cap C_i$  is included in  $B'_2 \cap C_i$ . Because this holds for every component  $C_i$  of the partition, we have  $B'_1 \subseteq B'_2$ .  $\square$

**Theorem 4 (Commutativity)** For all belief states  $B_1$  and  $B_2$ ,  $B_1 \otimes B_2 = B_2 \otimes B_1$ .

*Proof:* Directly by the symmetry of the definition.  $\square$

We define for belief states  $B$  and sets  $S$  of belief states

$$S \otimes B = B \otimes S = \bigcup \{B \otimes B' \mid B' \in S\}.$$

**Theorem 5 (Associativity)** For all belief states  $B_1, B_2$  and  $B_3$ ,  $(B_1 \otimes B_2) \otimes B_3 = B_1 \otimes (B_2 \otimes B_3)$ .

*Proof:* We show that every belief state in  $(B_1 \otimes B_2) \otimes B_3$  is also in  $B_1 \otimes (B_2 \otimes B_3)$ . Because of the commutativity of  $\otimes$ , showing that belief states in  $B_1 \otimes (B_2 \otimes B_3) = (B_3 \otimes B_2) \otimes B_1$  are also in  $(B_1 \otimes B_2) \otimes B_3 = B_3 \otimes (B_2 \otimes B_1)$  is by exactly the same argument.

Take any belief state  $B$  in  $(B_1 \otimes B_2) \otimes B_3$ . Take any component  $C_i$  of the partition (with  $1 \leq i \leq n$ .) Now  $B \cap C_i$  equals either  $B_3 \cap C_i$  or  $(B_1 \otimes B_2) \cap C_i$ . In the latter case  $B \cap C_i$  equals either  $B_1 \cap C_i$  or  $B_2 \cap C_i$ .

Because  $B$  was generated by cartesian products of intersections of  $B_1, B_2$  and  $B_3$  with the observational classes, the intersections of  $B$  with the observational classes can be considered independently one at a time.

If  $B \cap C_i$  equals  $B_1 \cap C_i$ , then we have a belief state  $B'$  in  $B_1 \otimes (B_2 \otimes B_3)$  with  $B' \cap C_i = B \cap C_i$ .

If  $B \cap C_i$  equals  $B_2 \cap C_i$  (the case  $B_3 \cap C_i$  is the same because of commutativity), then there is a belief state  $B''$  in  $B_2 \otimes B_3$  with  $B'' \cap C_i = B \cap C_i$ , and further, a belief state  $B'$  in  $B_1 \otimes (B_2 \otimes B_3)$  with  $B' \cap C_i = B \cap C_i$ .  $\square$

Because the operator is associative and commutative, it unambiguously generalizes to sets of belief states as

$$\otimes \{B_1, B_2, \dots, B_n\} = B_1 \otimes B_2 \otimes \dots \otimes B_n.$$

This can be more directly expressed as

$$\otimes \{B_1, \dots, B_m\} = \{I_1 \cup \dots \cup I_n \mid \begin{array}{l} I_i \in \{B_1 \cap C_i, \dots, B_m \cap C_i\}, \\ I_i \not\subset B_j \cap C_i \text{ for } j \in \{1, \dots, m\} \\ \text{for all } i \in \{1, \dots, n\} \}. \end{array}$$

for a partition  $P = \langle C_1, \dots, C_n \rangle$  of the state space.

We use the generalized definition of the  $\otimes$  operation in implementing the planning algorithms. This is how we avoid the generation of a very high number of intermediate belief states that would otherwise be obtained by pairwise combinations of belief states.

**Theorem 6 (Inclusion)** For all  $B \in T$  there is  $B' \in \otimes T$  such that  $B \subseteq B'$ .

*Proof:* We construct  $B' = \bigcup_{i=1}^n I_i$  by constructing  $I_i$  and showing that  $B \cap C_i \subseteq I_i$  for every  $i$ .

Consider  $C_i$ . If  $B \cap C_i \subset B^* \cap C_i$  for no  $B^* \in T$ , then we choose  $I_i = B \cap C_i$ , and clearly  $B \cap C_i \subseteq I_i$ . If  $B \cap C_i \subset B^* \cap C_i$  for some  $B^* \in T$ , let  $B^*$  be such a belief state in  $T$  so that  $B^* \cap C_i$  is set-inclusion maximal. Now we choose  $I_i = B^* \cap C_i$ , and clearly  $B \cap C_i \subseteq I_i$ .

Because the inclusion  $B \cap C_i \subseteq I_i$  holds for all components of the partition,  $B \subseteq \bigcup_{i=1}^n I_i = B'$ .  $\square$

## Implementation of $\otimes$ with BDDs

Instead of representing individual states and belief states as lists of and lists of lists of atomic propositions, for many types of planning it is more efficient to represent sets of states implicitly as formulae. A computationally effective representation of propositional formulae and Boolean functions is binary decision diagrams (BDDs) (Bryant 1992). BDDs allow equivalence testing in constant time and many operations on Boolean functions in polynomial time. They have been widely applied in model-checking in computer-aided verification, and in the last years also in AI planning, especially in planning under uncertainty.

In this section we show how any two belief states  $B_1$  and  $B_2$  represented as BDDs can be combined to  $B_1 \otimes B_2$  by using standard operations on BDDs. Especially interesting is the use of the existential quantification operation of BDDs in handling partial observability. This BDD implementation of  $\otimes$  is the basis of our planning algorithm implementation.

The implementation of  $\otimes$  with BDDs is often much more efficient than a more direct implementation for two reasons. First, not all of the states in the belief states have to be represented separately. This is in general the main benefit of

BDDs. Second, the  $\otimes$  operation can be performed without iterating over every observational class (component of the partition), because BDDs very effectively allow handling those observational classes that intersect the belief states so that the intersections are in the set-inclusion relation.

The computation proceeds as follows.

1. Compute the observational classes that intersect both belief states so that the intersections are not in inclusion relation. This is by the following BDD computation.

$$X = \exists U(B_1 \wedge \neg B_2) \wedge \exists U(B_2 \wedge \neg B_1)$$

Here  $U$  is the set of unobservable state variables, and  $\exists$  denotes the existential abstraction operation on BDDs that is defined for one variable  $x$  as

$$\exists x \Phi \stackrel{\text{def}}{=} \Phi[0/x] \vee \Phi[1/x].$$

In the fully observable case  $X$  is the empty set, and in the unobservable case  $X$  is the universal set (if neither of the belief states subsumes the other.)

2. For those observational classes that intersect the belief states and the intersections are in the inclusion relation we can always choose the bigger intersection. The set of all those observational classes can directly be identified by simple BDD operations, and the explicit generation of these observational classes is avoided. The part of the state space that intersects both belief states so that the intersections are in the inclusion relation is simply

$$B = (B_1 \cup B_2) \setminus X.$$

3. For identifying the no-inclusion observational classes we iterate over the *cubes* of  $X$ , which are the disjuncts of a DNF of  $X$ . For every cube we assign the observable variables not occurring in the cube truth-values in all possible ways, in each case obtaining one observational class. The benefit of iterating over the cubes of  $X$  instead of using all the valuations of the observable variables is that  $X$  may include only a fraction of all observational classes. There are procedures for efficiently doing iteration over the cubes for example in the CUDD BDD package.
4. For every observational class  $C_i$  in  $X$  compute  $B_1^i = B_1 \cap C_i$  and  $B_2^i = B_2 \cap C_i$ .
5. Produce the  $2^n$  belief states as

$$B \cup B_{i_1}^1 \cup B_{i_2}^2 \cup \dots \cup B_{i_n}^n$$

where the  $i_j$  are assigned 1 or 2 in all  $2^n$  possible ways.

## Planning Algorithms

In this section we propose two algorithms for planning with partial observability that use the  $\otimes$  operation. The first algorithm – given in Figure 2 – exhaustively computes sets  $D_i$  for  $i \geq 0$  that contain all maximal belief states at distance  $i$  to the goal belief state.

In the algorithm description  $\text{preimg}_o(B)$  computes the strong preimage of a set  $B$  of states with respect to an operator  $o$  (Cimatti, Roveri, & Traverso 1998). The strong preimage is the maximal set of states from which a state in  $B$  is

*PROCEDURE* exhaustive()

```

i := 0;
D0 = {G};
WHILE I ⊆ B for no B ∈ Di and Di ≠ Di-1
  Di+1 := ⊗({preimgo(B) | B ∈ Di, o ∈ O} ∪ Di);
  i := i + 1;
END;
IF I ⊆ B for some B ∈ Di THEN plan has been found;

```

Figure 2: Algorithm that systematically generates the belief space

*PROCEDURE* heuristic()

```

i := 0;
D0 = {G};
A = D0;
WHILE I ⊆ B for no B ∈ Di and A ≠ ∅;
  B := an element of A of maximum cardinality;
  A := A \ {B};
  Di+1 := ⊗({preimgo(B) | o ∈ O} ∪ Di);
  A := A ∪ (Di+1 \ Di);
  i := i + 1;
END;
IF I ⊆ B for some B ∈ Di THEN plan has been found;

```

Figure 3: Algorithm that heuristically selects which belief states to expand

always reached by applying  $o$ . For deterministic operators this coincides with the standard (weak) preimage computation used in model-checking and other applications of BDDs (Burch *et al.* 1994). Both preimage computations can easily be implemented with the standard operations on BDDs.

The variables  $I$  and  $G$  are respectively the initial and the goal belief states, and the set  $O$  consists of the operators in the problem. This algorithm description assumes a uniform observability at all points of time so that only one operator  $\otimes$  that uses one partition of the state space to observational classes is needed. For the case in which the current observational classes depend on the operator last applied, we have to consider different functions  $\otimes$ , and to compute the preimage of a belief state only with respect to an operator that corresponds to the observability assumption with which the belief state was obtained by  $\otimes$ .

The second algorithm – given in Figure 3 – uses a heuristic for selecting one belief state at a time for preimage computation. Because the belief space is not traversed systematically, the correspondence between the sets  $D_i$  and the belief states at distance  $i$  is lost, and therefore distance information for the generated belief states has to be maintained separately. The heuristic tried with this second algorithm simply uses the cardinality of a belief state as the usefulness measure. On more complicated problems this is likely not to yield as good results as on the problems considered in the experiments described later in the paper.

After the initial belief state has been reached by using this backward computation, a branching plan can be extracted, as will be discussed in the next section.









