

# Observability and sensing (June 27, 2005)

## Motivation

### Conditional plans

- Definition
- Execution graphs

### Algorithms

#### AND-OR search forward

#### Backward search

- Branching
- Backward step
- Example
- Belief spaces
- Complexity
- Procedure *findnew*
- Main procedure

## Conditional plans

- A **conditional plan** is essentially a finite automaton (a graph).
- The nodes in the graph represent all the relevant information from earlier observations.
- For the **reachability** and **maintenance** objectives this information could just as well be represented by the **belief state**, and plans could be in principle defined also as mappings from belief states to actions. (This is, however, not sufficient for some more general objectives.)

## Conditional plans

### Execution

- Plan execution starts from a node  $n \in N$  and state  $s$  such that  $\langle \phi, n \rangle \in b$  and  $s \models I \wedge \phi$ .
- Execution in node  $n$  with  $l(n) = \langle o, B \rangle$ :
  - Execute  $o$ .
  - If  $\phi$  is true in all possible current states for some  $\langle \phi, n' \rangle \in B$  then continue execution from  $n'$ .  
At most one  $\phi$  may be true for this to be well-defined. Plan execution ends when none of the branch labels matches the current state. In a terminal node plan execution necessarily ends.
- Definition of **execution graphs** has to take into account the plan node: the nodes of the execution graphs are pairs  $(s, n)$  where  $s$  is a state of the transition system and  $n$  is a plan node.
- There is an edge from  $(s, n)$  to  $(s', n')$  if executing the action for plan node  $n$  in  $s$  may lead to state  $s'$  and node  $n'$ .

## Execution graphs of conditional plans

### Definition (Execution graph of a conditional plan)

Let  $\langle A, I, O, G, V \rangle$  be a transition system and  $\pi = \langle N, b, l \rangle$  a plan. The **execution graph** of  $\pi$  is  $\langle M, E \rangle$  where

- $M = S \times (N \cup \{\perp\})$  where  $S$  is all valuations of  $A$ ,
- $E \subseteq M \times M$  with an edge from  $\langle s, n \rangle$  to  $\langle s', n' \rangle$  iff
  - $l(n) = \langle o, B \rangle$  and
  - for some  $\langle \phi, n' \rangle \in B$   $s' \in \text{img}_o(s)$  and  $s' \models \phi$ .  
There is an edge from  $\langle s, n \rangle$  to  $\langle s', \perp \rangle$  iff
    - $l(n) = \langle o, B \rangle$ ,
    - $s' \in \text{img}_o(s)$ , and
    - there is no  $\langle \phi, n' \rangle \in B$  such that  $s' \models \phi$ .

The **initial nodes** are  $\langle s, n \rangle$  such that  $s \models I$  and  $s \models \phi$  for some  $\langle \phi, n \rangle \in b$ .

The **goal nodes** are  $\langle s, n \rangle$  such that  $s \models G$ .

# Planning with partial observability

- If not all can be observed, plans cannot be defined as mappings from states to actions because current state is not in general known.
- If something can be observed, plans cannot be defined as sequences of actions because different observations suggest different actions.
- A more general notion of plans is needed that generalizes both action sequences and state-to-action mappings.

## Conditional plans

### Definition

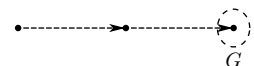
Let  $\Pi = \langle A, I, O, G, V \rangle$  be a succinct transition system. A **plan** for  $\Pi$  is a triple  $\langle N, b, l \rangle$  where

- $N$  is a finite set of nodes,
- $b \subseteq \mathcal{L} \times N$  maps initial states to starting nodes, and
- $l : N \rightarrow O \times 2^{\mathcal{L} \times N}$  is a function that assigns each node  $n$  an operator and a set of pairs  $\langle \phi, n' \rangle$  where  $\phi$  is a formula over the observable state variables  $V$  and  $n' \in N$  is a successor node. Nodes  $n$  with  $l(n) = \langle o, \emptyset \rangle$  for some  $o \in O$  are **terminal nodes**.

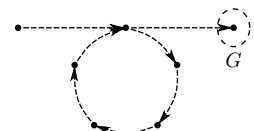
## Conditional plans

### Example

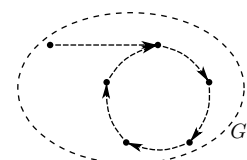
Bounded Reachability



Unbounded Reachability



Maintenance



## Summary of objectives

## Mapping memoryless plans to conditional plans

### Definition

Let  $\langle A, I, O, G, V \rangle$  be a transition system. Let  $S$  be the set of all states on  $A$ . Let  $\pi : S \rightarrow O$  be a memoryless plan. Define  $C(\pi) = \langle N, b, l \rangle$  where

1.  $N = O$ ,
2.  $b = \{ \langle FMA(\{s \in S | \pi(s) = o\}), o \rangle | o \in O \}$ , and
3.  $l(o) = (o, \{ \langle FMA(\{s \in S | \pi(s) = o'\}), o' \rangle | o' \in O \})$  for all  $o \in O$ .

Above  $FMA(T)$  is a formula  $\phi$  such that  $T = \{s \in S | s \models \phi\}$ . The memoryless plan  $\pi$  corresponds the conditional plan  $C(\pi)$  in the sense that the subgraphs induced by the initial nodes are isomorphic, and this isomorphism preserves both initial and goal nodes.

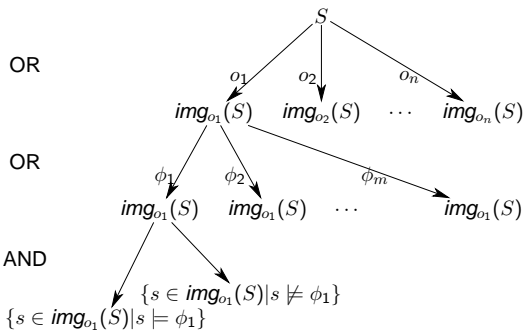
## Planning with partial observability

Algorithms

- ▶ **Heuristic (forward) search with AND-OR trees.**  
AO\* (plans without loops), LAO\* (with loops)
- ▶ **Dynamic programming (backward search)**  
Start from the set of goal states.  
Find state sets from which already generated state sets can be reached by actions and branching.
- ▶ **Reduction to full observability**  
For reachability and maintenance goals the planning problem is in principle solvable by **reduction** to fully observable planning in the **belief space**. But this is impractical because there are  $2^n$  belief states for  $n$  states, and  $2^{2^m}$  belief states for  $m$  state variables.

## Conditional planning with and-or search

Example



For  $n$  observable state variables there are  $m = \frac{2^n - 2}{2}$  non-trivial formulae to consider for **binary branching**.

## Backward search algorithms

- ▶ Flavor similar to the backward algorithms for fully observable problems.
- ▶ Backward steps with operator applications: strong preimages.
- ▶ Backward steps with branching: strong preimage of the union of belief states from different observational classes.

## Sufficiency of memoryless plans for full observability

## Conditional planning with and-or search

### AND-OR trees for conditional planning

- OR nodes 1 Choice of action
- OR nodes 2 Choice of observations
- AND nodes Nondeterminism (actions / observations)

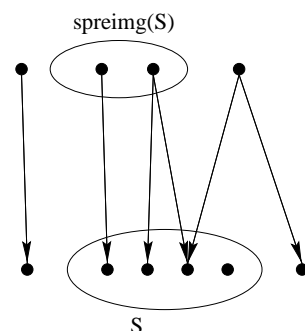
### Binary branching vs. general branching

Conditional plans can be defined with **binary branching** (IF-THEN-ELSE) or with **n-ary branching** (CASE/SWITCH). Latter can always be reduced to former.

## Conditional planning with and-or search

- ▶ Conflict between plan size and branching:
  1. If all observations are always used, plans become very big.
  2. If not all observations are used it may be impossible to find a plan.
 Trying out all possible ways to branch is not feasible. No good general solutions to this problem exist.
- ▶ AND-OR search algorithms use heuristics for making branching decisions.

## Regression/preimages

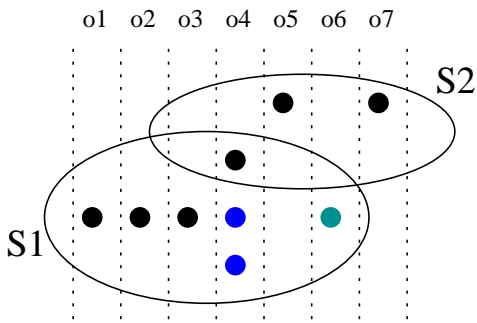


## Branching in backward search

- ▶ Let the observational classes be  $C_1, \dots, C_n$ .
- ▶ Let  $B_1, B_2, \dots, B_n$  be sets of states with plans so that for all  $i, j$  such that  $i \neq j$  there is no observational class  $C \in \{C_1, \dots, C_n\}$ , such that  $S_i \cap C \neq \emptyset$  and  $S_j \cap C \neq \emptyset$ .  
Now they can be combined to  $B = B_1 \cup \dots \cup B_n$  that has a plan starting with a branch.
- ▶ We may pick exactly one belief  $B_i$  state from every observational class.

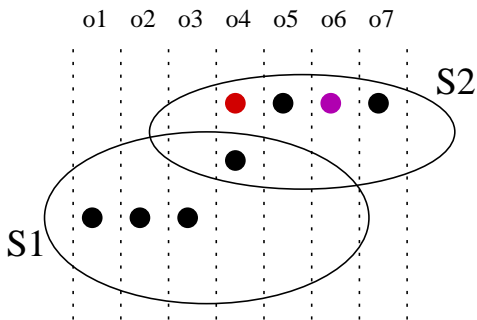
### Combination 11

4 observational classes with choice between plan plan for  $S_1$  or  $S_2$



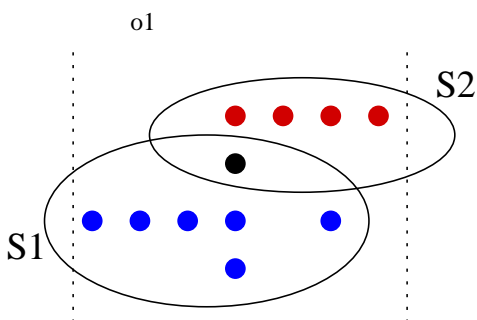
### Combination 22

4 observational classes with choice between plan plan for  $S_1$  or  $S_2$

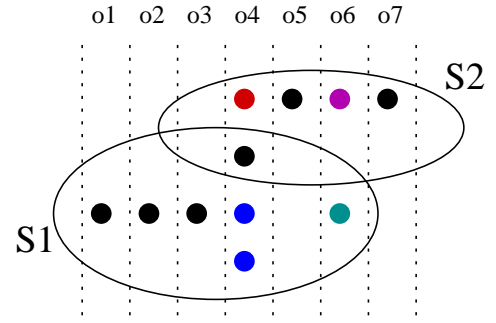


### No observability $\Rightarrow$ No branching

Only one observational class: no choice between subplans

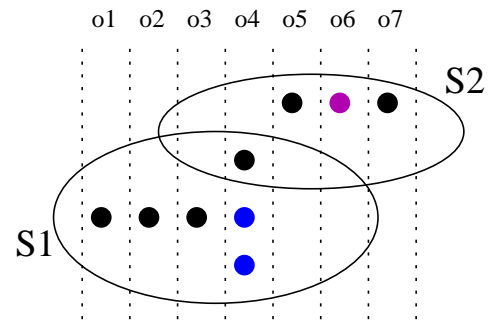


## Branching



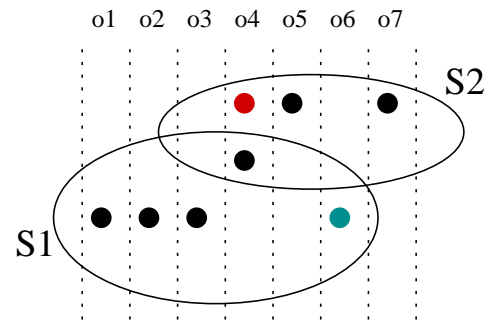
### Combination 12

4 observational classes with choice between plan plan for  $S_1$  or  $S_2$



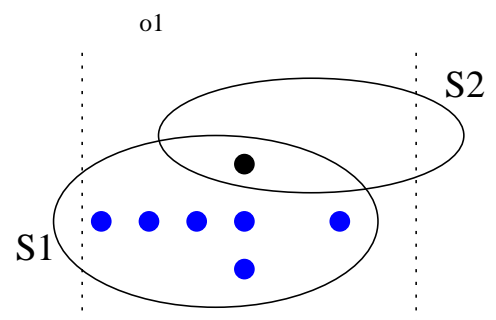
### Combination 21

4 observational classes with choice between plan plan for  $S_1$  or  $S_2$



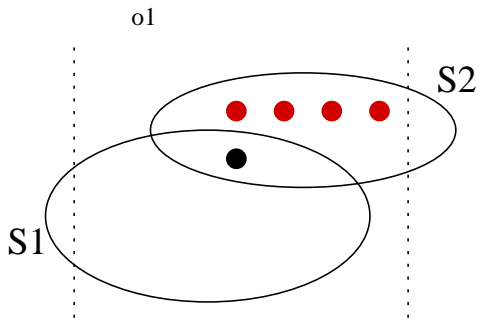
### Combination 1

No choice between subplans during execution



## Combination 2

No choice between subplans during execution



## One step in the backward algorithm

1. Pick from each observational class one belief state.
2. Compute the strong preimage of their union w.r.t. operator  $o$ .
3. Split the resulting set of states to belief states for different observational classes.

## Backward search in the belief space

Example

- ▶ Blocks world with 3 blocks
- ▶ Goal: all blocks are on the table
- ▶ Only the variables  $clear(X)$  are observable.
- ▶ A block can be moved onto the table if the block is clear.
- ▶ 8 observational classes corresponding to the 8 valuations of  $\{clear(A), clear(B), clear(C)\}$  (one of the valuations does not correspond to a blocks world state.)

## Data structure

### Definition (Belief space)

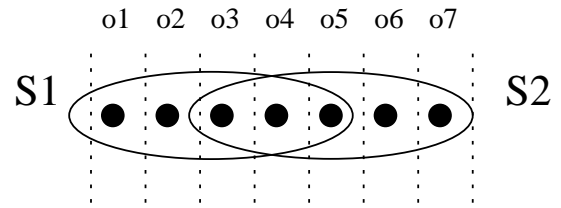
Let  $P = (C_1, \dots, C_n)$  be a partition of the set of all states. Then a *belief space* is an  $n$ -tuple  $\langle G_1, \dots, G_n \rangle$  where  $G_i \subseteq 2^{C_i}$  for all  $i \in \{1, \dots, n\}$  and  $B \not\subseteq B'$  for all  $i \in \{1, \dots, n\}$  and  $\{B, B'\} \subseteq G_i$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

A belief space is a set of belief states partitioned to subsets corresponding to the observational classes.

The simplest belief spaces are obtained from sets  $B$  of states as  $\mathcal{F}(B) = \{\{C_1 \cap B\}, \dots, \{C_n \cap B\}\}$ .

## Combination with full observability

Different plan can be used for every state



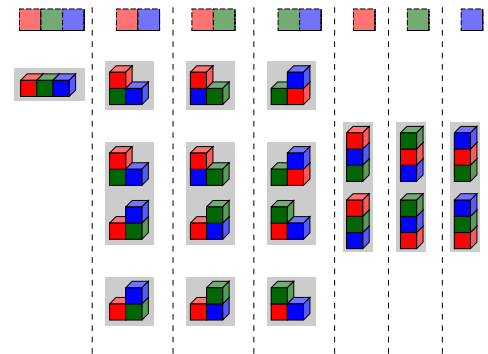
## Algorithm idea: construction of plans

If plans for belief states  $Z_1, \dots, Z_n$ , respectively corresponding to observational classes  $C_1, \dots, C_n$ , are  $\pi_1, \dots, \pi_n$ , then a plan for a new belief state is

1. Apply  $o$ .
2. If new current state is in  $C_i$  for  $i \in \{1, \dots, n\}$ , continue with  $\pi_i$ .

## Plan construction by backward search

Example: backward step with blue-block-onto-table



## Data structure

A belief space is extended as follows.

### Definition (Extension)

Let  $P = (C_1, \dots, C_n)$  be the partition of all states,  $G = \langle G_1, \dots, G_n \rangle$  a belief space, and  $T$  a set of states. Define  $G \oplus T$  as  $\langle G_1 \uplus (T \cap C_1), \dots, G_n \uplus (T \cap C_n) \rangle$  where the operation  $\uplus$  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  $U \uplus V = \{R \in U \cup \{V\} \mid R \not\subseteq K \text{ for all } K \in U \cup \{V\}\}$ .

## Data structure

A factored belief space  $G = \langle G_1, \dots, G_n \rangle$  can be viewed as representing the set of sets of states

$$\text{flat}(G) = \{s_1 \cup \dots \cup s_n \mid s_i \in G_i \text{ for all } i \in \{1, \dots, n\}\}.$$

The cardinality of  $\text{flat}(G)$  is  $|G_1| \cdot |G_2| \cdot \dots \cdot |G_n|$ .

(Albert-Ludwigs-Universität Freiburg)

AI Planning

June 27, 2005 33 / 39

Backward search Complexity

## Complexity of finding new belief states

The basic backward step in algorithms for partial observability is computationally difficult.

**Theorem**

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

**Proof**

Membership in NP is easy: nondeterministically choose  $s_i \in G_i$  for every  $i \in \{1, \dots, n\}$ , compute the preimage  $r$  of  $s_1 \cup \dots \cup s_n$ , verify that  $r \cap C_i$  for some  $C_i$  is not in  $G_i$ .

(Albert-Ludwigs-Universität Freiburg)

AI Planning

June 27, 2005 35 / 39

Backward search Procedure findnew

## An algorithm for finding new belief states

A new belief state can be found by the following algorithm that runs in worst-case exponential time.

```

1: PROCEDURE findnew( $o, A, F, H$ );
2: IF  $F = \langle \rangle$  AND  $\text{preimg}_o(A) \not\subseteq B$  for all  $B \in \text{flat}(H)$ 
3: THEN RETURN  $A$ ; (* New belief state was found *)
4: IF  $F = \langle \rangle$  THEN RETURN  $\emptyset$ ;
5:  $F$  is  $\langle \{f_1, \dots, f_m\}, F_2, \dots, F_k \rangle$  for some  $k \geq 1$ ;
6: FOR  $i := 1$  TO  $m$  DO
7:    $B := \text{findnew}(o, A \cup f_i, \langle F_2, \dots, F_k \rangle, H)$ ;
8:   IF  $B \neq \emptyset$  THEN RETURN  $B$ ;
9: END;
10: RETURN  $\emptyset$ ;
```

(Albert-Ludwigs-Universität Freiburg)

AI Planning

June 27, 2005 37 / 39

Backward search Main procedure

## Summary

- ▶ Planning with **partial observability** in general requires a definition of plans that generalizes plans respectively required in the **fully observable** and **unobservable** special cases: mappings state  $\rightarrow$  action and sequences of actions.
- ▶ Main algorithms:
  1. Reduction to full observability by viewing **belief states** as states.
  2. AND-OR search forward.
  3. Generation of belief states backward starting from goal belief states.

(Albert-Ludwigs-Universität Freiburg)

AI Planning

June 27, 2005 39 / 39

## Data structure: membership test

**Theorem**

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

**Proof.**

This is simply by testing whether for all  $i \in \{1, \dots, n\}$  there is  $B' \in G_i$  such that  $B \cap C_i \subseteq B'$ . □

(Albert-Ludwigs-Universität Freiburg)

AI Planning

June 27, 2005 34 / 39

Backward search Complexity

## Complexity of finding new belief states

**Proof continues.**

NP-hardness is by reduction from SAT. We illustrate the proof by an example. Let  $T = \{A \vee B \vee C, \neg A \vee B, \neg C\}$ .

Construct a belief space so that  $T$  is satisfiable iff strong preimage of  $o(x) = x_0$  is not in the FBS: clause is mapped to the set of literals not in it; satisfying valuation = a new belief state.

$$\langle \{ \{\widehat{A}, \widehat{B}, \widehat{C}\}, \{A, \widehat{B}, C, \widehat{C}\}, \{A, \widehat{A}, B, \widehat{B}, C\} \}, \{ \{A_0\}, \{\widehat{A}_0\} \}, \{ \{B_0\}, \{\widehat{B}_0\} \}, \{ \{C_0\}, \{\widehat{C}_0\} \} \rangle$$

□

(Albert-Ludwigs-Universität Freiburg)

AI Planning

June 27, 2005 36 / 39

Backward search Main procedure

A planning algorithm:  $\text{plan}(I, O, G)$ ;

```

1: PROCEDURE plan( $I, O, G$ );
2:  $H := \mathcal{F}(G)$ ;
3: progress := true;
4: WHILE progress and  $I \not\subseteq I'$  for all  $I' \in \text{flat}(H)$  DO
5:   progress := false;
6:   FOR EACH  $o \in O$  DO
7:      $B := \text{findnew}(o, \emptyset, H, H)$ ;
8:     IF  $B \neq \emptyset$  THEN (* New belief state was found *)
9:       BEGIN
10:         $H := H \oplus \text{preimg}_o(B)$ ; (* Add it to belief space *)
11:        progress := true;
12:      END;
13:    END;
14:  END;
15: IF  $I \subseteq I'$  for some  $I' \in \text{flat}(H)$  THEN RETURN true;
16: ELSE RETURN false;
```

(Albert-Ludwigs-Universität Freiburg)

AI Planning

June 27, 2005 38 / 39