Lecture 14: Planning with unobservability

- Unobservable problems

- Algorithms for planning without observability: regression, BDDs, QBF

- Computational complexity of planning without observability
Belief space: example
Belief space: example II

A robot without any sensors anywhere in the classroom.

Actions: go North, South, East, West

Plan: $6 \times$ West, $7 \times$ North, $1 \times$ East, $1 \times$ North
Belief space: example II, belief state initially
Belief space: belief state after $W$
Belief space: after WW

door

[Diagram of a grid with a shaded area and a marked door]

Jussi Rintanen

December 2, AI Planning 6/48
Belief space: after WWW

![Belief space diagram](image-url)
Belief space: after WWWWW

door

---

Jussi Rintanen

December 2, AI Planning  8/48
Belief space: after WWWWW

door

[Grid diagram]

December 2, AI Planning  9/48
Belief space: after WWWWWW
Belief space: after WWWWWWWN
Belief space: after WWWWWWWNNN

Jussi Rintanen

December 2, AI Planning 12/48
Belief space: after WWWWWWWWNNN
Belief space: after WWWWNNNNN
Belief space: after WWWWWWWWNNNNNN

door
Belief space: after WWWWNNNNNN
Belief space: after WWWWWWWNNNNNNN

door

Jussi Rintanen

December 2, AI Planning
Belief space: after WWWWWWWNNNNNNNNE

door

Jussi Rintanen

December 2, AI Planning 18/48
Belief space: after WWWWWWWNNNNNNNEN
Planning under unobservability

Find a path from $I$ to a belief state $S \subseteq G$.

There is edge from $Z$ to $Z'$ if for some $o \in O$

1. $o$ is applicable in all states in $Z$.
2. $\text{img}_o(Z) = Z'$.

Equivalently, $Z = \text{spreimg}_o(Z')$ for some $o \in O$.

We will define regression for nondeterministic operators next...
Regression for nondeterministic operators

1. Nondeterministic operator \( o = \langle c, e \rangle \).

2. Transform the effect \( e \) into normal form II:

\[
p_1 e_1 | \cdots | p_n e_n
\]

where every \( e_i \) is deterministic and in NF.

3. \( \text{regr}_{\langle c, e \rangle}(\phi) \) is now defined as \( \text{regr}_{\langle c, e_1 \rangle}(\phi) \land \cdots \land \text{regr}_{\langle c, e_n \rangle}(\phi) \).

THEOREM. Let \( S' = \{ s' | s' \models \phi \} \). Then \( \text{spreimg}_{o}(S') = \{ s | s \models \text{regr}_{o}(S') \} \).
Regression: example

\[ o = \langle A, (0.5B|0.5\neg C) \rangle \]

\[
\text{regr}_o(B \leftrightarrow C) = \text{regr}_{\langle A,B \rangle}(B \leftrightarrow C) \land \text{regr}_{\langle A,\neg C \rangle}(B \leftrightarrow C) \\
= (A \land (\top \leftrightarrow C')) \land (A \land (B \leftrightarrow \bot)) \\
= (A \land C') \land (A \land \neg B) \\
= A \land C \land \neg B
\]
Algorithms for unobservable planning I

Regression + heuristic search algorithms (A∗, ...) just as in deterministic planning

- new problem: Testing \( I \models \text{regr}_o(\phi) \) is co-NP-hard.
- If only one initial state, distance estimation as in deterministic planning can be used, but even accurate estimates for deterministic planning may be very misleading.
- Good methods for estimating distances between belief states not known.
Algorithms for unobservable planning II

images (fwd) / strong preimages (bwd), BDDs and heuristic search (A*)

- problem: BDDs may grow much faster than formulae with regression.
- heuristic 1: distances from the BDD-based FO algorithms
- heuristic 2: size of the current set of states
  - backward search: try to increase cardinality of belief state
  - forward search: try to decrease cardinality of belief state
- Good heuristics not known.
Algorithms for unobservable planning III

Translation into quantified Boolean formulae (QBF)

Why not by translation into propositional logic?

• We need to be able to say that there is a plan s.t. ...  
  This is like the satisfiability problem in CPC: there is a valuation...

• We need to be able to say that for all executions ...  
  This is like the validity problem in CPC: for all valuations...
Quantified Boolean formulae extend the language of propositional logic with quantifiers:

If $\phi$ is a propositional formula and $\sigma$ is a sequence of $\exists p$ and $\forall p$, one for every $p \in P$, then $\sigma \phi$ is a QBF.

A formula $\exists x \phi$ is true if and only if $\phi[\top/x] \lor \phi[\bot/x]$ is true. (Equivalently, $\phi[\top/x]$ is true or $\phi[\bot/x]$ is true.)

A formula $\forall x \phi$ is true if and only if $\phi[\top/x] \land \phi[\bot/x]$ is true. (Equivalently, $\phi[\top/x]$ is true and $\phi[\bot/x]$ is true.)
A formula $\phi$ with an empty prefix (and consequently without occurrences of propositional variables) is true if and only if $\phi$ is satisfiable (equivalently, valid: for formulae without propositional variables validity coincides with satisfiability.)

**EXAMPLE.**

The formulae $\forall x \exists y (x \leftrightarrow y)$ and $\exists x \exists y (x \land y)$ are true.

The formulae $\exists x \forall y (x \leftrightarrow y)$ and $\forall x \forall y (x \lor y)$ are false.
UO planning with QBF

There is a sequence of operators so that for all executions that start in an initial state reach a goal state.

\[ \exists o_1 \cdots o_m \cdots o_1^t \cdots o_n^t \]
\[ \forall p_1^0 \cdots p_n^0 a_{\sigma_1}^1 \cdots a_{\sigma_k}^1 \cdots a_{\sigma_1}^t \cdots a_{\sigma_k}^t \]
\[ \exists p_1^1 \cdots p_n^1 \cdots p_1^t \cdots p_n^t \]
\[ (I^0 \rightarrow (\mathcal{R}_3^0(P^0, P^1) \land \cdots \land \mathcal{R}_3^{t-1}(P^{t-1}, P^t) \land \Gamma^t)) \]

Variables \(a_{ij}^t\) are for encoding nondeterministic effects.
UO planning with QBF: nondeterminism

\[ o_7 = \langle A, (0.3(A \land (B \triangleright D))|0.3(B \land C)|0.4C) \rangle \]

\( \lceil \log_2 3 \rceil = 2 \) auxiliary variables \( a_{\sigma,0}, a_{\sigma,1} \) for 3 alternatives

<table>
<thead>
<tr>
<th>valuation</th>
<th>effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \neg a_{\sigma,1} \land \neg a_{\sigma,0} )</td>
<td>( A \land (B \triangleright D) )</td>
</tr>
<tr>
<td>( \neg a_{\sigma,1} \land a_{\sigma,0} )</td>
<td>( B \land C )</td>
</tr>
<tr>
<td>( a_{\sigma,1} \land \neg a_{\sigma,0} )</td>
<td>( C )</td>
</tr>
<tr>
<td>( a_{\sigma,1} \land a_{\sigma,0} )</td>
<td>( C )</td>
</tr>
</tbody>
</table>

Here \( \sigma \) is a unique id for the nondeterministic choice.
UO planning with QBF: nondeterminism, nesting

\[(0.5A|0.5B) \land (0.5((0.5C|0.5D) \land E)|0.5F)\]

\[
\begin{align*}
&\sigma = o, 1 \\
&\sigma = o, 2, 1 \\
&\sigma = o, 2
\end{align*}
\]

\[-a_{o, 1} \quad A \\
a_{o, 1} \quad B \\
-a_{o, 2} \land -a_{o, 2, 1} \quad C \\
-a_{o, 2} \land a_{o, 2, 1} \quad D \\
-a_{o, 2} \quad E \\
a_{o, 2} \quad F
\]
UO planning with QBF: effects, precons

\[(o_7 \land \neg a_{7,1} \land \neg a_{7,0}) \rightarrow A'
\]
\[(o_7 \land \neg a_{7,1} \land \neg a_{7,0} \land B) \rightarrow D'
\]

\[(o_7 \land \neg a_{7,1} \land a_{7,0}) \rightarrow B'
\]
\[(o_7 \land \neg a_{7,1} \land a_{7,0}) \rightarrow C''
\]

\[(o_7 \land a_{7,1} \land \neg a_{7,0}) \rightarrow C'
\]
\[(o_7 \land a_{7,1} \land a_{7,0}) \rightarrow C''
\]

\[o_7 \rightarrow A\]
UO planning with QBF: frame axioms

\[\neg A \land A' \rightarrow ((o_7 \land \neg a_{7,1} \land \neg a_{7,0}) \lor \cdots)\]
\[A \land \neg A' \rightarrow \cdots\]
\[\neg B \land B' \rightarrow ((o_7 \land \neg a_{7,1} \land a_{7,0}) \lor \cdots)\]
\[B \land \neg B' \rightarrow \cdots\]
\[\neg C \land C' \rightarrow ((o_7 \land a_{7,1} \land \neg a_{7,0}) \lor (o_7 \land a_{7,1} \land a_{7,0}) \lor \cdots)\]
\[C \land \neg C' \rightarrow \cdots\]
\[\neg D \land D' \rightarrow ((o_7 \land B \land \neg a_{7,1} \land \neg a_{7,0}) \lor \cdots)\]
\[D \land \neg D' \rightarrow \cdots\]
UO planning with QBF: $\mathcal{R}_3^i(P_i, P_{i+1})$

The formula $\mathcal{R}_3^i(P_i, P_{i+1})$ is then the conjunction of all the formulae for

- effects of operators,
- preconditions of operators,
- frame axioms for all the state variables,
- $\neg o \lor \neg o'$ for pairs of interfering operators $o$ and $o'$

similarly to the encoding of deterministic planning.
UO planning with QBF

Properties of the translation into QBF.

- Efficient for short plans with a lot of parallelism.
- Inefficient for long plans with little parallelism.
- Better algorithms for evaluating QBF would help...
Complexity of unobservable planning

THEOREM 1. The problem of testing the existence of a plan for problem instances without observability is EXPSPACE-hard.

Proof idea: Simulate deterministic Turing machines with an exponential space bound.

THEOREM 2. The problem of testing the existence of a plan for problem instances without observability is in EXPSPACE.

Proof idea: adapt the proof of PSPACE-membership of deterministic planning.
EXPSPACE-hardness of UO planning

Problem: exponentially many tape cells!! Representing them all requires exponentially many state variables. Reduction not polynomial time!!

Solution: Probabilistic test whether the plan describes an execution of the TM.

1. For every execution randomly choose a watched tape cell.

2. Check that the plan correctly represents the watched tape cell. (Because of unobservability the plan cannot “see” what the watched tape cell is and it always has to simulate correctly.)
EXPSPACE-hardness of UO planning

Let $\langle \Sigma, Q, \delta, q_0, g \rangle$ be any deterministic Turing machine with an exponential space bound $e(x)$.

Let $\sigma$ be an input string of length $n$. We denote the $i$th symbol of $\sigma$ by $\sigma_i$.

The Turing machine may use space $e(n)$, and for encoding numbers from 0 to $e(n) + 1$ corresponding to the tape cells we need $m = \lceil \log_2(e(n) + 2) \rceil$ bits.
EXPSPACE-hardness of UO planning

The set \( P \) of state variables in the problem instance consists of

1. \( q \in Q \) for denoting the internal states of the TM,

2. \( w_i \) for \( i \in \{0, \ldots, m - 1\} \) for representing the watched tape cell \( j \in \{0, \ldots, e(n) + 1\} \),

3. \( s \in \Sigma \cup \{|, □\} \) for the contents of the watched tape cell,

4. \( h_i \) for \( i \in \{0, \ldots, m - 1\} \) for representing the position of the R/W head \( j \in \{0, \ldots, e(n) + 1\} \).
EXPSPACE-hardness of UO planning

The uncertainty in the initial state is about which tape cell is the watched one.

$I$ is the conjunction of the following formulae.

1. $q_0$

2. $\neg q$ for all $q \in Q \setminus \{q_0\}$.

3. Formulae for having the contents of the watched tape cell in
state variables $\Sigma \cup \{|, \Box\}$.

$$|
\leftrightarrow (w = 0)
$$

$$\Box \leftrightarrow (w > n)
$$

$$s \leftrightarrow \bigvee_{i \in \{1, \ldots, n\}, \sigma_i = s} (w = i) \text{ for all } s \in \Sigma
$$

4. $h = 1$ for the initial position of the R/W head.

$w = i, w > i$ denote the formulae for testing integer equality and inequality of the numbers encoded by $w_0, w_1, \ldots$.

Later we use effects $h := h + 1$ and $h := h - 1$ that increment and decrement the number encoded by $h_0, h_1, \ldots$.
EXPSPACE-hardness of UO planning

The goal is the following formula.

\[ G = \bigvee \{ q \in Q \mid g(q) = \text{accept} \} \]
EXPSPACE-hardness of UO planning

For all \(\langle s, q \rangle \in (\Sigma \cup \{\|, \square\}) \times Q\) and \(\langle s', q', m \rangle \in (\Sigma \cup \{\|\}) \times Q \times \{L, N, R\}\) define the effect \(\tau_{s,q}(s', q', m)\) as \(\alpha \land \kappa \land \theta\).

Meaning of \(\alpha\), \(\kappa\) and \(\theta\) like in the EXPTIME-hardness proof of conditional planning with FO:

\(\alpha = \) effect for change in current tape cell
\(\kappa = \) effect for change in state of TM
\(\theta = \) effect for tape movement
EXPSPACE-hardness of UO planning, $\alpha$

$\alpha$ describes what happens to the tape symbol under the R/W head.

$\alpha = \top$ if the tape cell does not change, i.e. $s = s'$.

Otherwise,

$$\alpha = (h = w) \triangleright (\neg s \land s')$$

to change the watched tape cell.
EXPSPACE-hardness of UO planning, $\kappa$

$\kappa$ describes the change to the internal state of the TM.

$\kappa = \top$ if $q = q'$

$\kappa = \neg q \land q'$ if $q \neq q'$ and movement $\neq R$

$\kappa = \neg q \land ((h < e(n)) \blacktriangleright q')$ movement is $R$ and $q \neq q'$

The condition $h < e(n)$ prevents reaching an accepting state if the space bound is violated.
EXPSPACE-hardness of UO planning, $\theta$

$\theta$ describes the movement of the R/W head. Either there is movement to the left, no movement, or movement to the right. Hence

$$\theta = \begin{cases} 
  h := h - 1 & \text{if } m = L \\
  \top & \text{if } m = N \\
  h := h + 1 & \text{if } m = R 
\end{cases}$$
EXPSPACE-hardness of UO planning

Consider \( \langle s, q \rangle \in \Sigma \times Q \). If \( g(q) = \exists \) and \( \delta(s, q) = \langle s', q', m \rangle \), then define

\[ o_{s,q} = \langle ((h \neq w) \lor s) \land q \land (h \leq e(n)), \tau_{s,q}(s', q', m) \rangle \]

The condition \((h \neq w) \lor s\) is the key to the EXPSPACE-hardness proof: If the plan tries to cheat here, then the operator is not applicable on some execution, and the plan cannot reach the goals.
EXPSPACE-hardness of UO planning

We claim that the problem instance has a plan if and only if the Turing machine accepts without violating the space bound.

If the simulation of the Turing machine violates the space bound, then $h > e(n)$ and a goal state cannot be reached because no further operator will be applicable.
EXPSPACE-hardness of UO planning

\[
\begin{array}{c|c|c}
q_1 & 12122\Box & o_{1,q_1} \\
q_2 & 22122\Box & o_{2,q_2} \\
q_7 & 23122\Box & o_{1,q_7} \\
q_3 & 23222\Box & o_{2,q_3} \\
q_2 & 23232\Box & o_{2,q_2} \\
q_4 & 23132\Box & o_{2,q_4} \leftarrow \text{first cheating here!} \\
q_4 & 21132\Box & o_{2,q_4}
\end{array}
\]