Lecture 4

- Computation of distance estimates
- Symmetries in transition systems
Distance estimation for heuristic search

- Computing the distance between a state and a set of states is PSPACE-hard (and hence also NP-hard.)

- Compute for $i = 1, 2, 3, \ldots$ what values each state variable can (possibly) have after applications of $i$ operators. (Irrespective of other state variables.)

- The word “possibly” above refers to admissibility of the function $h(n)$ that is obtained: we may underestimate the distances, but never overestimate.
Distance estimation: functions $v_i$

For each point of time $i \geq 0$ valuation $v_i : P \rightarrow \{\{0\}, \{1\}, \{0, 1\}\}$.

Meaning: $p$ may have any of the truth-values in $v_i(p) \subseteq \{0, 1\}$ after $i$ operator applications.

$v_0(p) = \{I(p)\}$: The initial state gives the possible values at 0.

$v_i(p) = v_{i-1}(p) \cup v_i^{o1}(p) \cup \cdots \cup v_i^{on}(p)$ for every $i \geq 1$: Extend the sets with every applicable operator.

The functions $v_i^o$ will be described in a moment...
Distance estimation: possible truth-values of formulas

Based on possible truth-values of state variables, recursively compute possible truth-values of formulae.

<table>
<thead>
<tr>
<th>( \neg )</th>
<th>( \lor )</th>
<th>( \land )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( 0 )</td>
<td>( 0 )</td>
<td>( 0 )</td>
</tr>
<tr>
<td>( 1 )</td>
<td>( 1 )</td>
<td>( 1 )</td>
</tr>
<tr>
<td>{0, 1}</td>
<td>{0, 1}</td>
<td>{0, 1}</td>
</tr>
</tbody>
</table>
Distance estimation: possible truth-values of formulas

Let \( v(A) = \{0, 1\} \) and \( v(B) = \{1\} \).

\[
\widehat{v}(A \land B) = \{0, 1\} \\
\widehat{v}(A \lor B) = \{1\} \\
\widehat{v}((A \land B) \land (A \lor B)) = \{0, 1\} \\
\widehat{v}(\neg A) = \{0, 1\} \\
\widehat{v}(A \land \neg A) = \{0, 1\} \\
\widehat{v}(B \land \neg B) = \{0\}
\]
Distance estimation: functions $v^o_i$

Extend $v_i$ with truth-values obtained with operator $o \in O$.

$$o = \langle z, (c_1 \triangleright p_1) \land (\overline{c_1} \triangleright \neg p_1) \land \cdots \land (c_n \triangleright p_n) \land (\overline{c_n} \triangleright \neg p_n) \rangle$$

If $\hat{v}_i(z) = \{0\}$, then $v^o_{i+1} = v_i$. (operator not applicable (yet))

Otherwise, for every $p \in P$ occurring in the effect,

$$v^o_{i+1}(p) = v^o_i(p) \cup \{1 | 1 \in \hat{v}_i(c)\} \cup \{0 | 1 \in \hat{v}_i(\overline{c})\}.$$  

And for every $p \in P$ not occurring in the effect, $v^o_{i+1}(p) = v_i(p)$. 

Jussi Rintanen

October 28, AI Planning

6/27
Distance estimation: example, distance 1 to 3
Distance estimation: example, distance 1 to 3

Distance estimates:

<table>
<thead>
<tr>
<th></th>
<th>$L_1$</th>
<th>$L_2$</th>
<th>$L_3$</th>
<th>$L_4$</th>
<th>$L_5$</th>
<th>$L_6$</th>
<th>$L_7$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$v_0$</td>
<td>{1}</td>
<td>{0}</td>
<td>{0}</td>
<td>{0}</td>
<td>{0}</td>
<td>{0}</td>
<td>{0}</td>
</tr>
<tr>
<td>$v_1$</td>
<td>{0, 1}</td>
<td>{0, 1}</td>
<td>{0}</td>
<td>{0}</td>
<td>{0}</td>
<td>{0, 1}</td>
<td>{0}</td>
</tr>
<tr>
<td>$v_2$</td>
<td>{0, 1}</td>
<td>{0, 1}</td>
<td>{0}</td>
<td>{0, 1}</td>
<td>{0, 1}</td>
<td>{0, 1}</td>
<td>{0}</td>
</tr>
<tr>
<td>$v_3$</td>
<td>{0, 1}</td>
<td>{0, 1}</td>
<td>{0, 1}</td>
<td>{0, 1}</td>
<td>{0, 1}</td>
<td>{0, 1}</td>
<td>{0, 1}</td>
</tr>
</tbody>
</table>

Estimate for 1 to 3 is 3. Also other estimates are accurate.
Distance estimation: example 2, distance 1 to 3

Jussi Rintanen  
October 28, AI Planning  9/27
Distance estimation: example 2, distance 1 to 3

Distance estimates:

<table>
<thead>
<tr>
<th></th>
<th>$B_2$</th>
<th>$B_1$</th>
<th>$B_0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$v_0$</td>
<td>{0}</td>
<td>{0}</td>
<td>{1}</td>
</tr>
<tr>
<td>$v_1$</td>
<td>{0, 1}</td>
<td>{0, 1}</td>
<td>{0, 1}</td>
</tr>
<tr>
<td>$v_2$</td>
<td>{0, 1}</td>
<td>{0, 1}</td>
<td>{0, 1}</td>
</tr>
</tbody>
</table>

Distance estimate for 1 to 3 is 1. Actually, all states (except 1) have distance estimate 1.
Distance estimation: admissibility

Let $S_i$ be the set of states with distance $\leq i$ from the initial state.

Now for every $s \in S_i$ and $p \in P$, $s(p) \in v_i(p)$.

So, if $s \models \phi$ for some state $s$ with distance $\leq i$, then $1 \in \hat{v}_i(\phi)$: the heuristic is admissible.

The converse does not hold: That $1 \in \hat{v}_i(\phi)$ does not mean that there were a state $s$ with distance $\leq i$ so that $s \models \phi$. 
Symmetries

- Regularities in the structure of transition graphs
- The repeating subgraph may be small, but the whole graph very big.
- Search space can be reduced by recognizing symmetries \(\Rightarrow\) identifying symmetries may have a very big payoff
Symmetries
Symmetries: impact of goal states on symmetry
Symmetries: impact of goal states on symmetry
Symmetries: equivalence classes of states

\[ [A] = \{A, D, G, J\} \]
\[ [B] = \{B, E, H, K\} \]
\[ [C] = \{C, F, I, L\} \]

Search space consists of these equivalence classes.

Each equivalence class is represented by one of its members.
Symmetries: forward planning

- The equivalence classes form *the quotient state space*.

- Implementation of forward search with symmetries:
  1. States are mapped to their equivalence classes.
  2. Each equivalence class is represented by its first element (⇒ ordering on the states). **Savings:** Equivalence classes may be very big, and we essentially consider only one state per equivalence class!!!
  3. Progression with an equivalence class is progression with any of its states.
Symmetries: recognizing symmetries

- symmetry ∼ automorphism (= isomorphism with itself) of a (coloured) graph

- Graph isomorphism is not known to be in P (nor NP-hard).
  But, algorithms for finding automorphisms run very fast on almost all graphs. (But, there may be very many automorphisms, and the graph may be huge.)

- More practical approach: Automorphisms can often directly be identified from schematic representations.
Symmetries: symmetries from schematic operators

Consider a set $O$ of schematic operators that are instantiated with a set $B$ of objects.

A subset $B' \subseteq B$ generates symmetries if

- no object in $B'$ explicitly occurs in $O$, and
- no object is mentioned in the goal formula $G$, or
- $G$ is schematic and no object in $B'$ is explicitly mentioned in it.
Symmetries: symmetries from schematic ops

Two states, described by sets of schematic literals on \( P = \{q_1(u_1, \ldots, u_n), \ldots, q_m(u_1, \ldots, u_n)\} \) are in the same equivalence class if one is obtained from the other by interchanging \( u \in B \) and \( u' \in B \).

EXAMPLE: \( X = \{a, b, c\} \).
\[
\begin{align*}
s_1 \models & \text{on}(a, b) \land \neg \text{on}(a, c) \land \neg \text{on}(b, a) \land \neg \text{on}(b, c) \land \neg \text{on}(c, a) \land \neg \text{on}(c, b) \\
s_2 \models & \text{on}(c, b) \land \neg \text{on}(c, a) \land \neg \text{on}(b, c) \land \neg \text{on}(b, a) \land \neg \text{on}(a, c) \land \neg \text{on}(a, b) \\
\end{align*}
\]
\( s_1 \) and \( s_2 \) are in the same equivalence class.
Symmetries: example

Objects \( X = \{\text{lamp1}, \text{lamp2}, \text{lamp3}, \text{lamp4}\} \)

Goal is \( \forall x \in X \neg \text{on}(x) \)

\( \text{switch-off}(x) = \langle \text{on}(x), \neg \text{on}(x) \rangle \)

Initial state: \( I \models \text{on}(\text{lamp1}) \land \text{on}(\text{lamp2}) \land \text{on}(\text{lamp3}) \land \text{on}(\text{lamp4}) \)

All objects in \( X \) are interchangeable.
Symmetries: example, the operators and goals

\[
\langle \text{on}(\text{lamp1}), \neg \text{on}(\text{lamp1}) \rangle \\
\langle \text{on}(\text{lamp2}), \neg \text{on}(\text{lamp2}) \rangle \\
\langle \text{on}(\text{lamp3}), \neg \text{on}(\text{lamp3}) \rangle \\
\langle \text{on}(\text{lamp4}), \neg \text{on}(\text{lamp4}) \rangle
\]

Goal is \( \neg \text{on}(\text{lamp1}) \land \neg \text{on}(\text{lamp2}) \land \neg \text{on}(\text{lamp3}) \land \neg \text{on}(\text{lamp4}) \)
Symmetries: example, the symmetries

For example the following states are symmetric:

\[ s_1 \models \neg \text{on}(lamp_1) \land \neg \text{on}(lamp_2) \land \text{on}(lamp_3) \land \text{on}(lamp_4) \]
\[ s_2 \models \text{on}(lamp_1) \land \neg \text{on}(lamp_2) \land \neg \text{on}(lamp_3) \land \text{on}(lamp_4) \]

In fact, there are 5 equivalence classes:

- class 1: all lamps are off (1 state)
- class 2: 1 lamp is on (4 states)
- class 3: 2 lamps are on (6 states)
- class 4: 3 lamps are on (4 states)
- class 5: 4 lamps are on (1 state)
Symmetries: example, the ordering on the states

Ordering on the state variables:
on(lamp1), on(lamp2), on(lamp3), on(lamp4)

Ordering on states lexicographically:
0000, 0001, 0010, 0011, 0100, 0101, 0110, 0111, 1000, 1001, 1010, 1011, 1100, 1101, 1110, 1111
Symmetries: example, plan search

Jussi Rintanen  
October 28, AI Planning  26/27
Symmetries: example, cont’d

1. Initial state is 1111.
2. Neighbors/successors of 1111 are 0111, 1011, 1101, 1110.
3. All in the same equivalence class, ignore all but 0111.
4. Neighbors/successors of 0111 are 0011, 0101, 0110.
5. All in the same equivalence class, ignore all but 0011.
6. Neighbors/successors of 0011 are 0001, 0010.
7. All in the same equivalence class, ignore all but 0001.
8. Neighbors/successors of 0001 are 0000.

At each step all but one action/state is ignored: search space size shrinks from $2^n$ to $n + 1$; number of plans from $n!$ to 1.