L07 – Non-Deterministic Domains

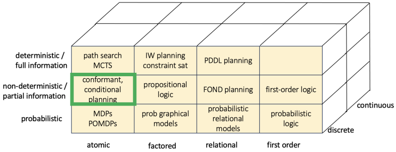
AIMA4e: Required: 4.3, 4.4

What you should know after this lecture

- We can design robust agents for non-deterministic domains
- If actions are non-deterministic (but observable), do and-or contingent planning
- If you don't know the state exactly, and won't get observations, do conformant planning in belief space
- If both things occur, you can do conditional planning in belief space!

Path-search problems with uncertainty

Atomic states
May only know an initial state <u>set</u>
Transitions may only specify a resulting state <u>set</u>



Vacuum world doesn't suck!

Example environment from AIMA:

- Robot in grid world has a vacuum cleaner
- Each cell in grid has dirt or not
- Robot can suck (with vacuum) or move left, right, up, or down
- Want the room to be completely clean

If completely observable and deterministic, then we can solve easily using min-cost-path search. But what if:

- The vacuum or the robot does not always work as expected?
- The robot does not know whether the grid cells are dirty (or potentially even where it is)?

Non-Deterministic Actions

This is sometimes known as the FOND (fully-observable non-deterministic) planning setting.

- Assume that the robot can always correctly observe the current world state
- But, when it takes an action, there is a set of possible outcomes.
- After each action, it can observe the result and decide what to do

Problem formulation:

- Given $(\mathcal{S}, \mathcal{A}, s_0, \tilde{T}, G, C)$ where $\tilde{T} : \mathcal{S} \times \mathcal{A} \rightarrow \text{Powerset}(\mathcal{S})$
- Find a contingent (conditional) plan in the form of a decision tree.
- Measuring the cost of a solution is tricky: could be maximum or average over the costs of the possible paths.

And/Or search

Search tree with alternating layers of node types:

- Or nodes : like our traditional search-tree nodes, where we get to pick an action
- And nodes : there are several possible resulting states and we have to find a plan for all of them.

Resulting plan is a tree with

- Internal nodes labeled with actions
- Branches labeled with states (that could possibly occur as a result of the action)
- Terminal leaf nodes indicating plan success.

There are optimal AO search methods (e.g. AO^*) but we will just look in detail at a simple one.

Depth-first And/Or search

AO-DFS$(s_0, (\mathcal{A}, \tilde{T}, G))$

return OR-SEARCH$(s_0, [\,], (\mathcal{A}, \tilde{T}, G))$

OR-SEARCH$(s, \textit{path}, (\mathcal{A}, \tilde{T}, G))$

if $s \in G$: **return** SUCCESS-LEAF()
if $s \in \textit{path}$: **return None** // Found a cycle
for $a \in \mathcal{A}$:
    $\textit{plan\_dict} =$ AND-SEARCH$(\tilde{T}(s, a), \textit{path} + [s], (\mathcal{A}, \tilde{T}, G))$
    if $\textit{plan\_dict} \neq$ **None**: **return** TREE-NODE$(a, \textit{plan_dict})$
return None

AND-SEARCH$(\textit{states}, \textit{path}, (\mathcal{A}, \tilde{T}, G))$

$\textit{plan\_dict} = \{\,\}$
for $s \in \textit{states}$:
    $\textit{plan} =$ OR-SEARCH$(s, \textit{path}, (\mathcal{A}, \tilde{T}, G))$
    if $\textit{plan} =$ **None**: **return None**
    $\textit{plan\_dict}[s] = \textit{plan}$
return $\textit{plan\_dict}$

What about cycles?

Sometimes, you just need cycles! Throw balls at target until you hit it!

- In line 2 of OR-SEARCH, return a special CYCLE-LEAF(s) node
- In execution, if you hit a CYCLE-LEAF(s), trace up your path to find state s and begin executing from there.

Non-observability

What if the robot does not know what state it is in and can't gain any information? We'll call this UOND (un-observable non-deterministic) planning.

- Could be uncertainty about the initial state
- Could be additional non-determinism in transitions that increase uncertainty
- If no possibility for observations, then we can try to find a conformant plan, which is guaranteed to succeed for all possible initial states and non-deterministic outcomes
- Do this through state-space search where now our states are belief states, which are sets of original states.

Reducing UOND planning to belief-space search

Given a non-observable-path problem $(\mathcal{S}, \mathcal{A}, \tilde{T}, G, C, S_o)$ where $\tilde{T}$ is non-deterministic as above and $S_0 \subset \mathcal{S}$ is the set of possible starting states, we can generate a standard min-cost-path problem $(\mathcal{B}, \mathcal{A}', T', G', C, b_0)$ so that solution to the min-cost-path problem is a solution to the original non-observable problem:

- $\mathcal{B} = \text{powerset}(\mathcal{S})$ // Set of subsets of $\mathcal{S}$
- $\mathcal{A}' = \mathcal{A}$
- $b_0 = S_0$ // Single belief state is a set of env states
- $T'(b, a) = \bigcup_{s \in b} T(s, a)$
- $G' = \{b \mid b \subseteq G\} = \text{powerset}(G)$
- $C'(b, a, b')$ can really only depend on a

Important to note that even though transition on states $\tilde{T}$ is non-deterministic, the belief-space transition function T' is deterministic.

Adding observations

If we can make some observations, then we get the POND (partially observable non-deterministic) planning setting.

- Finite observation set $\mathcal{O}$
- perception function $O : \mathcal{S} \to \mathcal{O}$

It tells us what we will see in each state. For example, a vacuum robot with a local dirt detector might have $\mathcal{O} = \{\mathit{clean}, \mathit{dirty}\}$ and perception function telling it about its current location.

Includes completely observable and completely unobservable cases. How?

Use it to do a belief update:

- Given current belief (set of possible states) S and an actual observation o, what should we believe?
- Rule out states that could not have generated o:
- $\textsc{Update}(b, o) = \{s \in b \mid O(s) = o\}$

Searching in PO environments

And/Or search in belief space! Have to do **and** branches on the possible observations. Given non-deterministic, partially observable problem $(\mathcal{S}, \mathcal{A}, \mathcal{O}, S_0, \tilde{T}, O, G, C)$ we can generate a non-deterministic observable problem (solvable by AO search):

- $\mathcal{B} = \text{powerset}(\mathcal{S})$ // Set of subsets of $\mathcal{S}$
- $\mathcal{A}' = \mathcal{A}$
- $b_0 = S_0$
- $\tilde{T}'(b, a) = \{\textsc{Update}(\tilde{T}(b, a), o) \mid o \in \textsc{possible-percepts}(T(b, a))\}$
- $G' = \{b \mid b \subseteq G\} = \text{powerset}(G)$
- $C(b, a, b')$ can really only depend on a

What observations could we possibly get in belief state b?

$$\textsc{possible-percepts}(b) = \{O(s) \mid s \in b\}$$

Agent for partially observable environment

- Initial belief $b = b_0$
- Make AO plan
- Take action a at root note
- Receive observation o from environment
- Update $b = \textsc{Update}(\tilde{T}(b, a), o)$
- Follow o branch in plan to get next action
- ...

Idea for later: In <u>receding-horizon</u> control, you can get away with making an approximate plan, and then replanning after every belief update.

Next time

- Propositional logic