Informatics 2D: Reasoning and Agents

Alex Lascarides

School of informatics

THE UNIVERSITY OF EDINBURGH

Lecture 18c: Planning and Acting in the Real World: Contingent Planning

Where are we?

Last time...

- Representing beliefs (to handle partially observable states)
- Sensorless planning

Today...

- Representing actions with nondeterminate outcomes
- Contingent planning

Conditional Effects

- So far, we have only considered actions that have the same effects on all states where the preconditions are satisfied.
- This means that any initial belief state that is a conjunction is updated by the actions to a belief state that is also a conjunction.
- But some actions are best expressed with conditional effects.
- This is especially true if the effects are non-deterministic, but in a bounded way.

Extending action representations

- Disjunctive effects: $Action(Left, \text{Precond:}AtR, \text{Effect:}AtL \vee AtR)$
- Conditional effects:

$$Action(Vacuum,\ \text{Precond:}\ \text{Effect:}(\textbf{when}\ AtL:\ CleanL) \wedge (\textbf{when}\ AtR:\ CleanR))$$

- Combination:

$$Action(Left,\ \text{Precond:}AtR\ \text{Effect:}AtL \vee (AtL \wedge (\textbf{when}\ CleanL:\ \neg CleanL)))$$

- Conditional steps: **if** $AtL \wedge CleanL$ **then** *Right* **else** *Vacuum*

The earlier painting furniture example (Lecture 18a)

Planning Problem

Table and chair, two cans of paint,
can open can, paint furniture with paint inside
Goal: table and chair the same colour

Contingent Plan

1. Look at the table and chair to sense their colours.
2. If they're the same colour, you're done.
3. If not, look at the paint cans.
4. If one of the can's is the same colour as one of the pieces of furniture, then apply that paint to the other piece of furniture.
5. Otherwise, paint both pieces with one of the cans.

The Three Actions (from Lecture 18a)

$$Action(RemoveLid(can),$$
$$\text{Precond:} Can(can)$$
$$\text{Effect:} Open(can))$$

$$Action(Paint(x, can),$$
$$\text{Precond:} Object(x) \wedge Can(can) \wedge Color(can, c) \wedge Open(can)$$
$$\text{Effect:} Color(x, c))$$

$$Action(LookAt(x),$$
$$\text{Precond:} InView(y) \wedge (x \neq y)$$
$$\text{Effect:} InView(x) \wedge \neg InView(y))$$

Percepts (from Lecture 18a)

$$Percept(Color(x, c),$$
$$\text{Precond:} Object(x) \wedge InView(x))$$

$$Percept(Color(can, c),$$
$$\text{Precond:} Can(can) \wedge Open(can) \wedge InView(can))$$

Formal Representation of the Contingent Plan

$[LookAt(Table), LookAt(Chair)$
  if $Color(Table, c) \wedge Color(Chair, c)$ **then** $NoOp$
    else $[RemoveLid(C_1), LookAt(C_1), RemoveLid(C_2), LookAt(C_2),$
      if $Color(Chair, c) \wedge Color(can, c)$ **then** $Paint(Table, can)$
      else if $Color(Table, c) \wedge Color(can, c)$ **then** $Paint(Chair, can)$
      else $[Paint(Chair, C_1), Paint(Table, C_1)]]]$

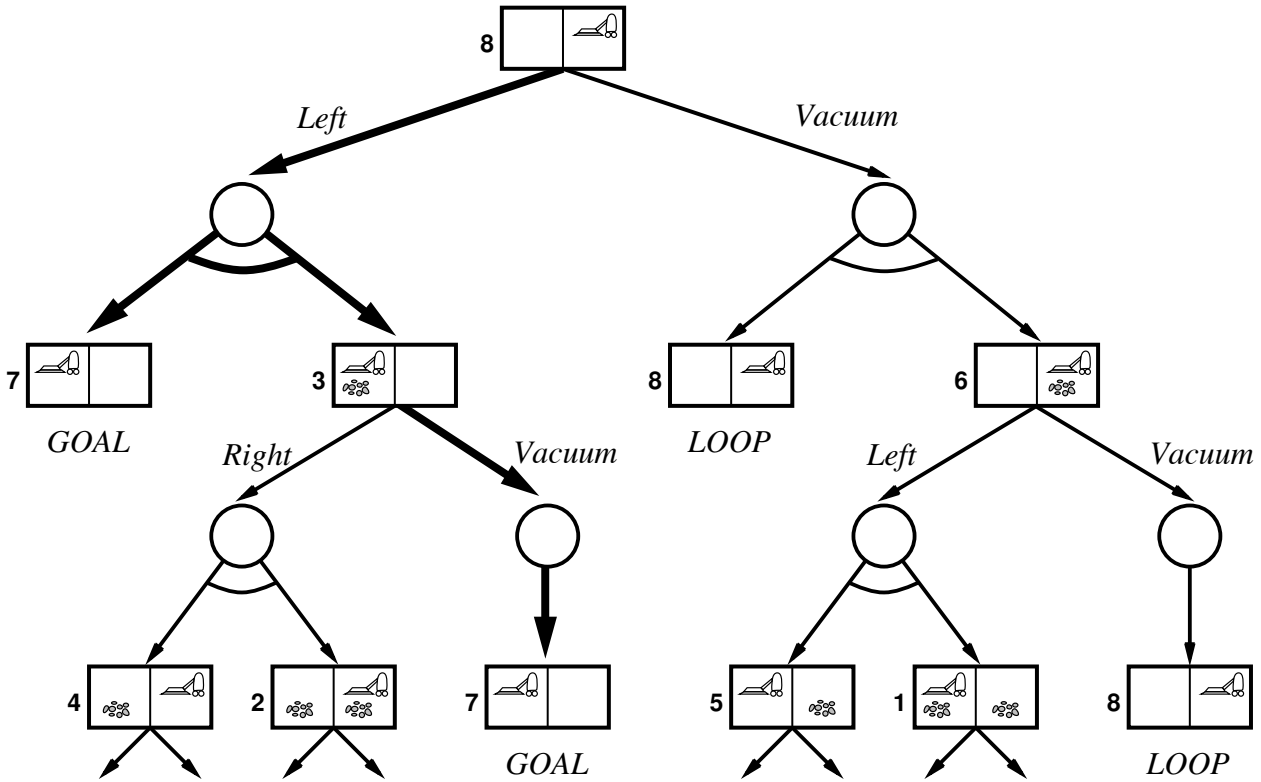
- Variables (e.g., c) are existentially quantified.

Games against nature

- Conditional plans should succeed regardless of circumstances
- Nesting conditional steps results in trees
- Similar to adversarial search, **games against nature**
- Game tree has state nodes and chance nodes where nature determines the outcome
- Definition of solution: A subtree with
  - a goal node at every leaf
  - specifies one action at each state node
  - includes every outcome at chance node
- AND-OR graphs can be used in similar way to the minimax algorithm (basic idea: find a plan for every possible result of a selected action)

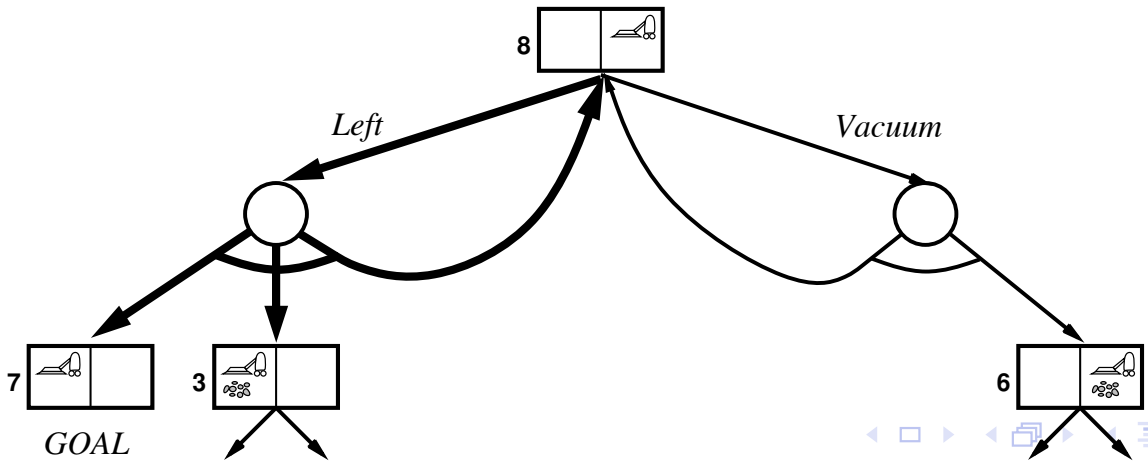
Example: "double Murphy" vacuum cleaner

- This wicked vacuum cleaner sometimes deposits dirt when moving to a clean destination or when vacuuming in a clean square
- Solution: [*Left*, **if** *CleanL*; **then** [] **else** *Vacuum*]



Acyclic vs. cyclic solutions

- If identical state is encountered (on same path), terminate with failure (if there is an acyclic solution it can be reached from previous incarnation of state)
- However, sometimes all solutions are cyclic!
- E.g., "triple Murphy" (also) sometimes fails to move.
- Plan [*Left*, **if** *CleanL* **then** [] **else** *Vacuum*] doesn't work anymore
- Cyclic plan:
  [*L* : *Left*, **if** *AtR* **then** *L* **elseif** *CleanL* **then** [] **else** *Vacuum*]

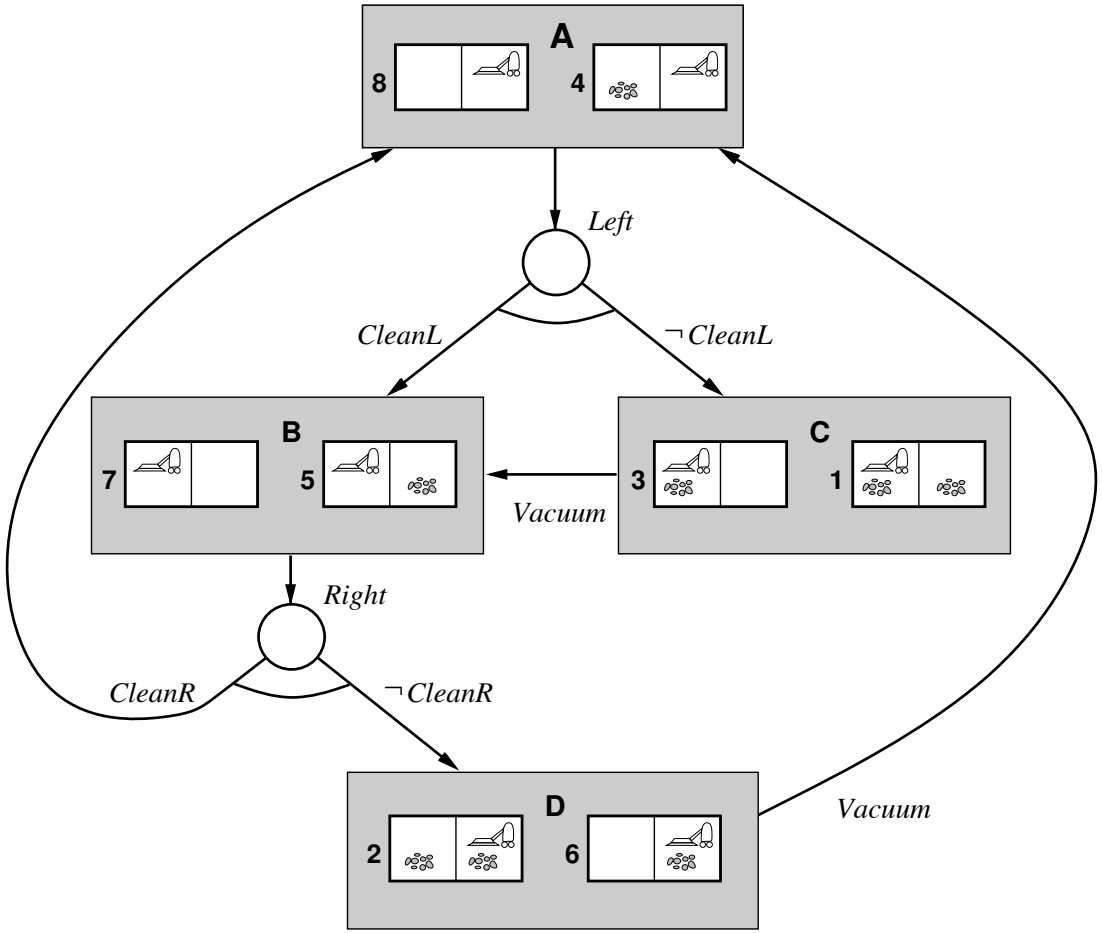


Nondeterminism and partially observable environments

"alternate double Murphy":

- Vacuum cleaner can sense cleanliness of square it's in, but not the other square, and
- dirt can sometimes be left behind when leaving a clean square.
- Plan in fully observable world: "Keep moving left and right, vacuuming up dirt whenever it appears, until both squares are clean and in the left square"
- But now goal test cannot be performed!

Housework in partially observable worlds



Conditional planning, partial observability

- Basically, we can apply our AND-OR-search to belief states (rather than world states)
- Full observability is special case of partial observability with singleton belief states
- Is it really that easy?
- Not quite, need to describe
  - representation of belief states
  - how sensing works
  - representation of action descriptions

Summary

- Methods for planning and acting in the real world
- Dealing with indeterminacy
- Contingent planning: use percepts and conditionals to cater for all contingencies.
- Fully observable environments: AND-OR graphs, games against nature
- Partially observable environments: belief states, action and sensing
- Next time: **Monitoring, re-planning, and Hierarchical Task Networks**