

today's topics:

- logic-based agents

## Logic-Based Agents

- When we started talking about logic, it was as a means of representing knowledge.
- We wanted to represent knowledge in order to be able to build agents.
- We now know enough about logic to do that.
- We will now see how a *logic-based agent* can be designed to perform simple tasks.
- Assume each agent has a *database*, i.e., set of FOL-formulae.  
These represent information the agent has about environment.

- We'll write  $\Delta$  for this database.
- Also assume agent has set of *rules*.  
We'll write  $R$  for this set of rules.
- We write  $\Delta \vdash_R \phi$  if the formula  $\phi$  can be proved from the database  $\Delta$  using only the rules  $R$ .
- How to program an agent:  
*Write the agent's rules  $R$  so that it should do action  $a$  whenever  $\Delta \vdash_R Do(a)$ .*  
Here,  $Do$  is a predicate.
- Also assume  $A$  is set of actions agent can perform.

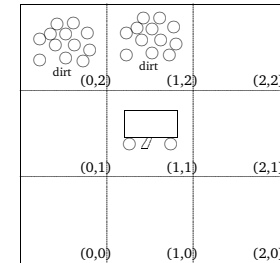
- The agent's operation:

1. for each  $a$  in  $A$  do
2.     if  $\Delta \vdash_R Do(a)$  then
3.         return  $a$
4.     end-if
5. end-for
6. for each  $a$  in  $A$  do
7.     if  $\Delta \not\vdash_R \neg Do(a)$  then
8.         return  $a$
9.     end-if
10. end-for
11. return *null*

- An example:

We have a small robot that will clean up a house. The robot has sensor to tell it whether it is over any dirt, and a vacuum that can be used to suck up dirt. Robot always has an orientation (one of *n*, *s*, *e*, or *w*). Robot can move forward one “step” or turn right 90°. The agent moves around a room, which is divided grid-like into a number of equally sized squares. Assume that the room is a 3 × 3 grid, and agent starts in square (0, 0) facing north.

- Illustrated:



- Three *domain predicates* in this exercise:

$In(x, y)$  agent is at  $(x, y)$   
 $Dirt(x, y)$  there is dirt at  $(x, y)$   
 $Facing(d)$  the agent is facing direction  $d$

- For convenience, we write rules as:

$$\phi(\dots) \longrightarrow \psi(\dots)$$

- First rule deals with the basic cleaning action of the agent

$$In(x, y) \wedge Dirt(x, y) \longrightarrow Do(suck) \quad (1)$$

- Hardwire the basic navigation algorithm, so that the robot will always move from (0, 0) to (0, 1) to (0, 2) then to (1, 2), (1, 1) and so on.

- Once agent reaches (2, 2), it must head back to (0, 0).

$$In(0, 0) \wedge Facing(north) \wedge \neg Dirt(0, 0) \longrightarrow Do(forward) \quad (2)$$

$$In(0, 1) \wedge Facing(north) \wedge \neg Dirt(0, 1) \longrightarrow Do(forward) \quad (3)$$

$$In(0, 2) \wedge Facing(north) \wedge \neg Dirt(0, 2) \longrightarrow Do(turn) \quad (4)$$

$$In(0, 2) \wedge Facing(east) \longrightarrow Do(forward) \quad (5)$$

- Other considerations:

- adding new information after each move/action;
- removing old information.

- Suppose we scale up to 10 × 10 grid?

## Summary

- This lecture covered two logic-related topics.
- First is covered mechanical theorem proving:
  - Pointed out some problems.
  - Suggested resolution as a solution.
- Next we looked at how logic might be used to program an agent.
  - Assumes we have a theorem prover.