Searching: Intro

CPSC 322 Lecture 4

January 11, 2006
Textbook §2.0 – 2.3

Lecture Overview

Agents and Representations

- Recall that an agent is something that acts in an environment
- The agent also receives observations about the environment
  - this could be observations from sensors such as cameras, laser rangefinders, etc.
  - can also include "observations" of the agent's own past actions
- In a deterministic environment, the agent can perfectly predict the outcome of an action
  - doesn't need sensors: just needs to remember its own past actions

The Table-Lookup Agent

- An agent can be thought of as a mapping from observations to the new action that the agent will take
- How should agents be constructed? One choice:
  - agent takes in the sequence of observations
  - agent looks up the correct action for this sequence of observations based on an internal representation (e.g., a table)
- Such an agent could indeed behave rationally. What's the problem?

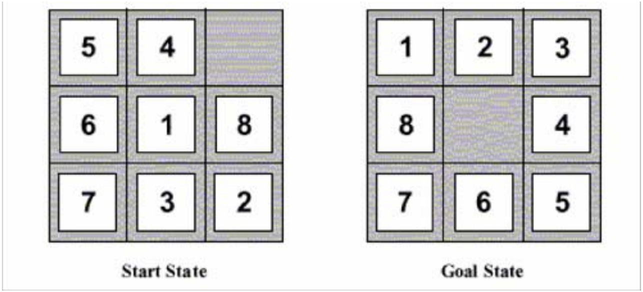
The Table-Lookup Agent

- An agent can be thought of as a mapping from observations to the new action that the agent will take
- How should agents be constructed? One choice:
  - agent takes in the sequence of observations
  - agent looks up the correct action for this sequence of observations based on an internal representation (e.g., a table)
- Such an agent could indeed behave rationally. What's the problem?
  - too many sequences of observations are possible!
  - e.g., 10 possible observations, 10 timesteps $\rightarrow 10^{10}$ different entries in the table
  - compare this to e.g., the number of different move sequences that are possible in chess

Example Problems

- To make things more concrete, let's think about some example problems:
  - solving a Sudoku
  - solving an 8-puzzle
  - the delivery robot planning the route it will take

What's an 8-Puzzle?



Example Problems

- To make things more concrete, let's think about some example problems:
  - solving a Sudoku
  - solving an 8-puzzle
  - the delivery robot planning the route it will take
- All of these problems are deterministic; thus, there's no need for any observations from sensors.
- Are these single or sequential decision problems?

Example Problems

- To make things more concrete, let's think about some example problems:
  - solving a Sudoku
  - solving an 8-puzzle
  - the delivery robot planning the route it will take
- All of these problems are deterministic; thus, there's no need for any observations from sensors.
- Are these single or sequential decision problems?
  - in fact, the distinction isn't really useful here; problems can be seen both ways
  - CSPs: settings where there's nothing meaningfully sequential about the decision
  - Planning: decisions are always sequential
  - Now: we're going to define the underlying tools that allow us to solve both

State Spaces

- Idea: sometimes it doesn't matter what sequence of observations brought the world to a particular configuration; it just matters how the world is arranged now.
- Represent the different configurations in which the world can be arranged as different states
  - which numbers are written in cells of the Sudoku and which are blank?
  - which numbers appear in which slots of the 8-puzzle?
  - where is the delivery robot?
- From each state, one or more actions may be available, which would move the world into a new state
  - write a new number in a blank cell of the Sudoku
  - slide a tile in the 8-puzzle
  - move the delivery robot to an adjacent location

Agent Design: trying again

- Let's update our table-based agent design around the idea of states
  - Now our agent maps from the given state to the chosen action
  - Our internal representation of this mapping is smaller:
    - sets of observations that lead to the same state are now represented only once in the table rather than many times
    - this can lead to exponential savings!
  - However, there's still a problem...

Agent Design: trying again

- Let's update our table-based agent design around the idea of states
  - Now our agent maps from the given state to the chosen action
  - Our internal representation of this mapping is smaller:
    - sets of observations that lead to the same state are now represented only once in the table rather than many times
    - this can lead to exponential savings!
  - However, there's still a problem... often, we don't understand the domain well enough to build the table
    - we'd need to be able to tell the agent how it should behave in every state
    - that's why we want intelligent agents: they should decide how to act for themselves
    - in order for them to do so, we need to give them goals

State Spaces

- Represent the different configurations in which the world can be arranged as different states
  - which numbers are written in cells of the Sudoku and which are blank?
  - which numbers appear in which slots of the 8-puzzle?
  - where is the delivery robot?
- From each state, one or more actions may be available, which would move the world into a new state
  - write a new number in a blank cell of the Sudoku
  - slide a tile in the 8-puzzle
  - move the delivery robot to an adjacent location
- Some states are goal states
  - A Sudoku state in which all numbers are different in each box, row and column
  - The single 8-puzzle state pictured earlier
  - The state in which the delivery robot is located in room 123

Search

- What we want to be able to do:
  - find a solution when we are not given an algorithm to solve a problem, but only a specification of what a solution looks like
  - idea: search for a solution
- What we need:
  - A set of states
  - A start state
  - A goal state or set of goal states
    - or, equivalently, a goal test: a boolean function which tells us whether a given state is a goal state
  - A set of actions
  - An action function: a mapping from a state and an action to a new state

Abstract Definition

How to search

- Start at the start state
- Consider the effect of taking different actions starting from states that have been encountered in the search so far
- Stop when a goal state is encountered

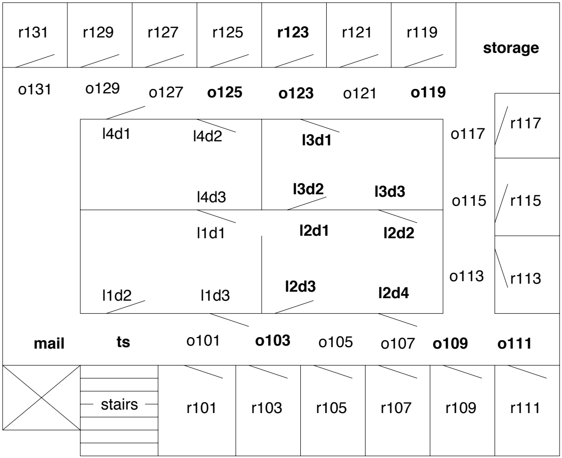
To make this more formal, we'll need to talk about graphs...

Search Graphs

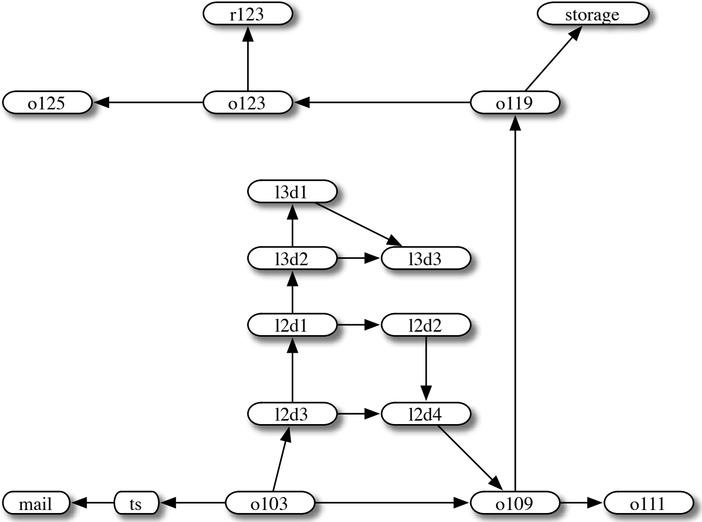
- A graph consists of
  - a set N of nodes;
  - a set A of ordered pairs of nodes, called arcs or edges.
- Node n_2 is a neighbor of n_1 if there is an arc from n_1 to n_2.
  - i.e., if $\langle n_1, n_2 \rangle \in A$
- A path is a sequence of nodes $\langle n_0, n_1, \ldots, n_k \rangle$ such that $\langle n_{i-1}, n_i \rangle \in A$.
- Given a start node and a set of goal nodes, a solution is a path from the start node to a goal node.

Example Domain for the Delivery Robot

The agent starts outside room 103,
and wants to end up inside room 123.



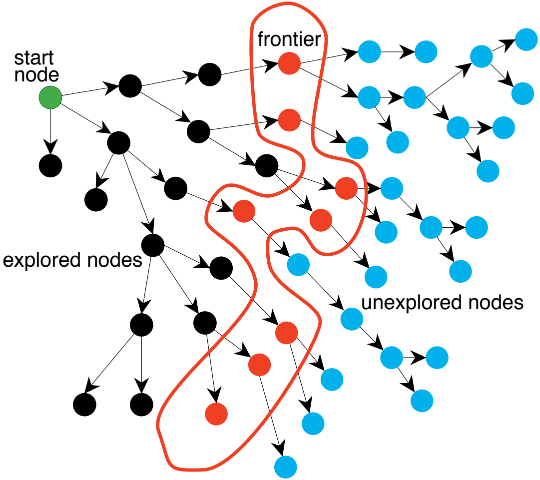
Example Graph for the Delivery Robot



Graph Searching

- Generic search algorithm: given a graph, start nodes, and goal nodes, incrementally explore paths from the start nodes.
- Maintain a frontier of paths from the start node that have been explored.
- As search proceeds, the frontier expands into the unexplored nodes until a goal node is encountered.

Problem Solving by Graph Searching



Graph Searching

- Generic search algorithm: given a graph, start nodes, and goal nodes, incrementally explore paths from the start nodes.
- Maintain a frontier of paths from the start node that have been explored.
- As search proceeds, the frontier expands into the unexplored nodes until a goal node is encountered.
- The way in which the frontier is expanded defines the search strategy.