1. Recap
2. Tree Search
3. Uniformed Search
   • Breath-First Search
   • Depth-First Search
   • Uni
Recap: Agents and Environments

Agent

Environment
Recap: Agents and Environments

An agent *perceives* its environment through *sensors* and *acts* upon it through *actuators*
Recap: Agents and Environments
Recap: Agents and Environments

Demo for Project 1, CSE 676 Deep Learning, Fall 2018 [Instructor: Dr. Sargur N. Srihari]
Authors: Alina Vereshchaka and Nathan Margaglio
Recap: Agents and Environments
Recap: Robot Taxi
Recap: Robot Taxi

Performance measures?
Recap: Robot Taxi

Performance measures:
Safe, fast, legal, comfortable trip, maximize profit
Recap: Robot Taxi

Performance measures:
Safe, fast, legal, comfortable trip, maximize profit

Environment?
Recap: Robot Taxi

Performance measures:
Safe, fast, legal, comfortable trip, maximize profit

Environment:
Roads, other traffic, pedestrians, customers
Recap: Robot Taxi

Performance measures:
Safe, fast, legal, comfortable trip, maximize profit

Environment:
Roads, other traffic, pedestrians, customers

Actuators?
Recap: Robot Taxi

Performance measures:
Safe, fast, legal, comfortable trip, maximize profit

Environment:
Roads, other traffic, pedestrians, customers

Actuators:
Steering accelerator, brake, signal, horn, display
Recap: Robot Taxi

Performance measures:  
Safe, fast, legal, comfortable trip, maximize profit

Environment:  
Roads, other traffic, pedestrians, customers

Actuators:  
Steering accelerator, brake, signal, horn, display

Sensors?
Recap: Robot Taxi

Performance measures:
Safe, fast, legal, comfortable trip, maximize profit

Environment:
Roads, other traffic, pedestrians, customers

Actuators:
Steering, accelerator, brake, signal, horn, display

Sensors:
Cameras, sonar, speedometer, GPS, keyboard, engine sensors
Recap: Intelligent Agents

What is the goal of the intelligent agent?
Recap: Intelligent Agents

Intelligent agents are supposed to maximize their performance measure.
State Tree Graphs

State space graph: A mathematical representation of a search problem
Nodes are (abstracted) world configurations
Arcs represent successors (action results)
The goal test is a set of goal nodes (maybe only one)

In a state space graph, each state occurs only once!

We can rarely build this full graph in memory (it’s too big), but it’s a useful idea
State Space Graphs

State space graph: A mathematical representation of a search problem
Nodes are (abstracted) world configurations
Arcs represent successors (action results)
The goal test is a set of goal nodes (maybe only one)

In a state space graph, each state occurs only once!

We can rarely build this full graph in memory (it’s too big), but it’s a useful idea
Search Trees

A search tree:
A “what if” tree of plans and their outcomes
The start state is the root node
Children correspond to successors
Nodes show states, but correspond to PLANS that achieve those states
For most problems, we can never actually build the whole tree
We construct both on demand – and we construct as little as possible.

Each NODE in in the search tree is an entire PATH in the state space graph.
Consider this 4-state graph:

How big is its search tree (from S)?
Consider this 4-state graph: How big is its search tree (from S)?
Tree Search
Goal formulation, based on the current situation and the agent’s performance measure,

Problem formulation is the process of deciding what actions and states to consider, given a goal.
Search Problems

A search problem consists of:

- A state space
- A successor function (with actions, costs)
- A start state and a goal test

A solution is a sequence of actions (a plan) which transforms the start state to a goal state
What’s in a State Space?

**PROBLEM:** Pathing

**STATES:** 

**ACTIONS:** 

**SUCCESSOR:** 

**GOAL TEST:**
What’s in a State Space?

PROBLEM: Pathing

STATES: (x, y) location

ACTIONS: NSEW

SUCCESSOR: update location only

GOAL TEST: is (x, y) = END
What’s in a State Space?

**PROBLEM:** Eat-All-Dots

**STATES:**

**ACTIONS:**

**SUCCESSOR:**

**GOAL TEST:**
What’s in a State Space?

PROBLEM: Eat-All-Dots

STATES: \{(x,y), \text{dot count}\}

ACTIONS: NSEW

SUCCESSOR: update location and possibly a dot count

GOAL TEST: dots all false
Problem-Solving Agent

function SIMPLE-PROBLEM-SOLVING-AGENT(percept) returns an action

persistent: seq, an action sequence, initially empty
state, some description of the current world state
goal, a goal, initially null
problem, a problem formulation

state ← UPDATE-STATE(state, percept)
if seq is empty then
  goal ← FORMULATE-GOAL(state)
  problem ← FORMULATE-PROBLEM(state, goal)
  seq ← SEARCH(problem)
  if seq = failure then return a null action
  action ← FIRST(seq)
  seq ← REST(seq)
return action
Node Components

For each node $n$ of the tree, we have a structure that contains four components:

- **n.State**: the state in the state space to which the node corresponds;
- **n.Parent**: the node in the search tree that generated this node;
- **n.Action**: the action that was applied to the parent to generate the node;
- **n.Path-Cost**: the cost, traditionally denoted by $g(n)$, of the path from the initial state to the node, as indicated by the parent pointers.
Search Algorithms

SEARCH ALGORITHMS

UNINFORMED SEARCH
“blind” Search
Have no additional information about the distance from current state to the goal

INFORMED SEARCH
Heuristic Search
Have additional information about the estimate distance from the current state to the goal
Search Algorithms

Scrolling Facebook feed vs Searching for weather
Search Algorithms

SEARCH ALGORITHMS

UNINFORMED SEARCH
- Depth First Search
- Breadth First Search
- Uniform Cost Search

INFORMED SEARCH
- Greedy Search
- A* Search
- Graph Search
DEPTH-FIRST SEARCH
Search Algorithms

SEARCH ALGORITHMS

UNINFORMED SEARCH
- Depth-First Search
- Breadth-First Search
- Uniform Cost Search

INFORMED SEARCH
- Greedy Search
- A* Search
- Graph Search
Depth-First Search

**Strategy:** expand a deepest node first

**Implementation:** Fringe is a LIFO stack
Depth-first search
Depth-first search
Depth-first search
Depth-first search
Depth-first search
Depth-first search
Depth-first search
BREADTH-FIRST SEARCH
Search Algorithms

SEARCH ALGORITHMS

UNINFORMED SEARCH
- Depth-First Search
- Breadth-First Search
- Uniform Cost Search

INFORMED SEARCH
- Greedy Search
- A* Search
- Graph Search
Breadth First Search

**Strategy:** expand a shallowest node first

**Implementation:** Fringe is a FIFO queue
Breadth-First Search

function BREADTH-FIRST-SEARCH(problem) returns a solution, or failure

node ← a node with STATE = problem.INITIAL-STATE, PATH-COST = 0
if problem.GOAL-TEST(node.STATE) then return SOLUTION(node)
frontier ← a FIFO queue with node as the only element
explored ← an empty set
loop do
  if EMPTY?(frontier) then return failure
  node ← POP(frontier) /* chooses the shallowest node in frontier */
  add node.STATE to explored
  for each action in problem.ACTIONS(node.STATE) do
    child ← CHILD-NODE(problem, node, action)
    if child.STATE is not in explored or frontier then
      if problem.GOAL-TEST(child.STATE) then return SOLUTION(child)
      frontier ← INSERT(child, frontier)
Breadth-First Search
Quiz: DFS vs BFS
BFS vs DFS

When will BFS outperform DFS?

When will DFS outperform BFS?
UNIFORM-COST SEARCH
Search Algorithms

SEARCH ALGORITHMS

UNINFORMED SEARCH
- Depth-First Search
- Breadth-First Search
- Uniform Cost Search

INFORMED SEARCH
- Greedy Search
- A* Search
- Graph Search
Uniform Cost Search

**Strategy:** expand a cheapest node first:

Fringe is a priority queue (priority: cumulative cost)
Uniform-Cost

Remember: UCS explores increasing cost contours

The good: UCS is complete and optimal!

The bad:
Explores options in every “direction”
No information about goal location

We’ll fix that soon!
Uniform-cost search

Problem: get from Sibiu to Bucharest
Uniform-cost search

Problem: get from Sibiu to Bucharest with least cost

Rimnicu Vilcea vs. Fagaras?
Uniform-cost search

Problem: get from Sibiu to Bucharest with least cost

Rimnicu Vilcea vs. Fagaras?
80 vs. 99
Uniform-cost search

**Problem:** get from Sibiu to Bucharest

Sibiu > Rimnicu Vilcea vs. Sibiu > Fagaras?
80 vs. 99

Rimnicu Vilcea > Pitesti: 80 + 97 = 177
Uniform-cost search

**Problem:** get from Sibiu to Bucharest

- Sibiu > Rimnicu Vilcea vs. Sibiu > Fagaras?
  - 80 vs. 99
- Rimnicu Vilcea > Pitesti: 80 + 97 = 177
- Sibiu > Fagaras: 99
Uniform-cost search

**Problem:** get from Sibiu to Bucharest

- Sibiu > Rimnicu Vilcea vs. Sibiu > Fagaras? 80 vs. 99
- Rimnicu Vilcea > Pitesti: $80 + 97 = 177$
- Sibiu > Fagaras: 99
- Fagaras > Bucharest: $99 + 211 = 310$
**Uniform-cost search**

**Problem:** Get from Sibiu to Bucharest

Sibiu > Rimnicu Vilcea vs. Sibiu > Fagaras?  
80 vs. 99

Rimnicu Vilcea > Pitesti: 80 + 97 = 177

Sibiu > Fagaras: 99

Fagaras > Bucharest: 99 + 211 = 310

Pitesti > Bucharest: 80 + 97 + 101 = 278