Search

Tuesday, February 2, 2021 9:23 AM

DFS, BFS (no cost calculation)
Uniform-cost search factors in cost

An agent must have an objective:
- Reflex action
- **Plan**: planning agents ask "what if", take decisions based on the outcome of the what if.
- Optimal and complete plan
- Replanning

Search problems:
- State space: start state, goal state, rest of the space
- Successor function

Solution is sequence of actions from start to goal (path)
- Path cost (optional)

State space:
- World state (the entire environment)
- Search state (abstraction of world state keeping only details needed)

Representation:
- Graphs: nodes are world configs, arcs are successor, goal is to reach goal state (each state happens only once)
- Search trees: start state -> possible futures -> possible future…->goal state (repetitive states, redundancy)

Open & Closed List
- Open: nodes that aren't yet visited
- Closed: nodes that are already visited

Search with search tree:
- Expand
- Maintain fringe (unexplored paths)
- Keep exploration minimum

Which fringe nodes should we explore? Algorithm should be
- Complete? Should find a solution if one exists
- Optimal? Least cost path
- Time complexity?
- Space Complexity?

branching factor
- 1 => b => b^2 => …b^m (m: max. depth

- Number of nodes in the entire tree: O(b^m)

DFS: use stack to maintain memory of parent nodes
- If m is finite: time = O(b^m)
- Fringe space: O(b*m)

Not optimal/complete

BFS: requires lot of space to store possibilities

Optimal since you are guaranteed the shallowest solution
Search time: O(b^s) (s: depth of first solution)
Requires FIFO queue to search fringe
S is finite, so optimal