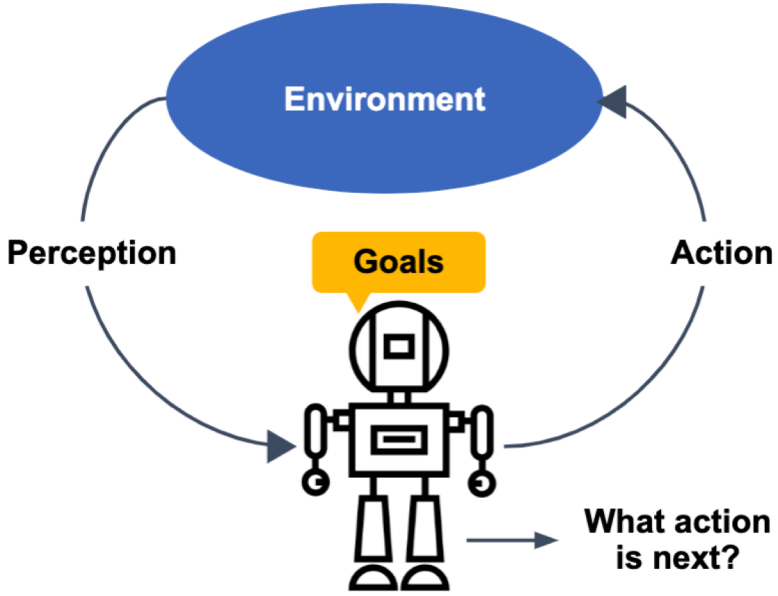
Sequential Decision-Making

Mehrdad Zakershahrak
Guest Speaker
Arizona State University

Ira A. Fulton Schools of Engineering
Arizona State University

Sequential Decision Making

- **Intelligent agency** involves **controlling the evolution** of external environments in **desirable ways**.
- Planning provides a way in which the agent can **maximize** its chances of effecting this control.
- Plan is **course of actions** that the agents decides upon based on its overall **goals**, information about the **current state** of the environment, and the **dynamics of its evolution**.



Classical Planning Problem

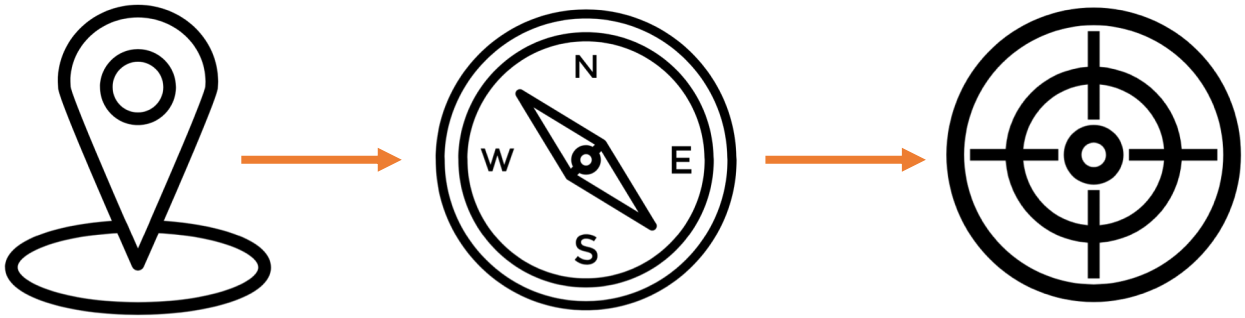
- The complexity of **plan synthesis** depends on a variety of **properties of the environment** and the agent.
- Perhaps the simplest case of planning occurs when the environment is **static**, (in that it changes only in response to the agents actions), **observable** and the agent's actions have **deterministic** effects on the state of the environment.
- Plan synthesis under these conditions has come to be known as the classical planning problem.

Classical Planning Problem

- Classical planning problem is still **computationally hard**, and has received a significant amount of attention.
- Work on classical planning has historically helped our understanding of planning under non-classical assumptions.

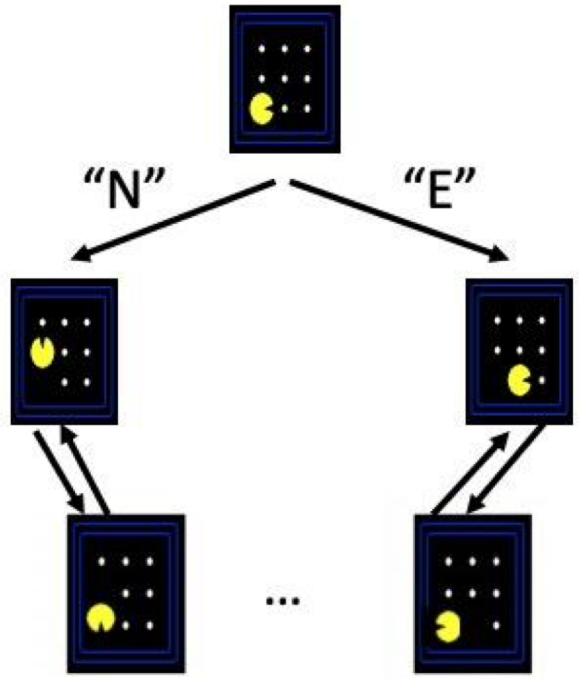
Example

- **Pathfinding**
  - States: (x,y) location
  - Actions: NSEW
  - Successor: updated location
  - Goal: is (x,y) = END?



Search Problem

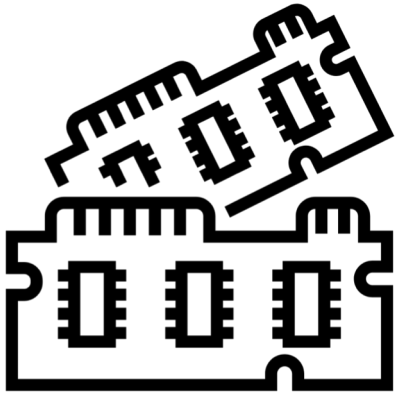
- **State space graph: A mathematical representation of search problem**
  - Each node are *abstracted* environment configurations
  - Edges are successors



Berkeley AI Materials. Date Accessed: June 17, 2019

Search Problem

- **Search tree**
  - a "What-if" tree of plans and their outcomes
  - initial state is the root
  - nodes correspond to states in the plan
  - children correspond to successors
- **Building the whole search tree takes lots of memory!**



Search Strategies

- **Order of node expansion matters**
- **Completeness**
  - Does it always find a solution if one exists?
- **Optimality**
  - Does it always find a least-cost solution?
- **Time complexity**
  - Number of nodes generated/expanded space
- **Space complexity**
  - Maximum number of nodes in memory

Time and Space Complexity

- **Time and space complexity are measured with:**
  - maximum branching factor
  - depth of the shallowest solution
  - maximum depth of the search space

Uninformed Search

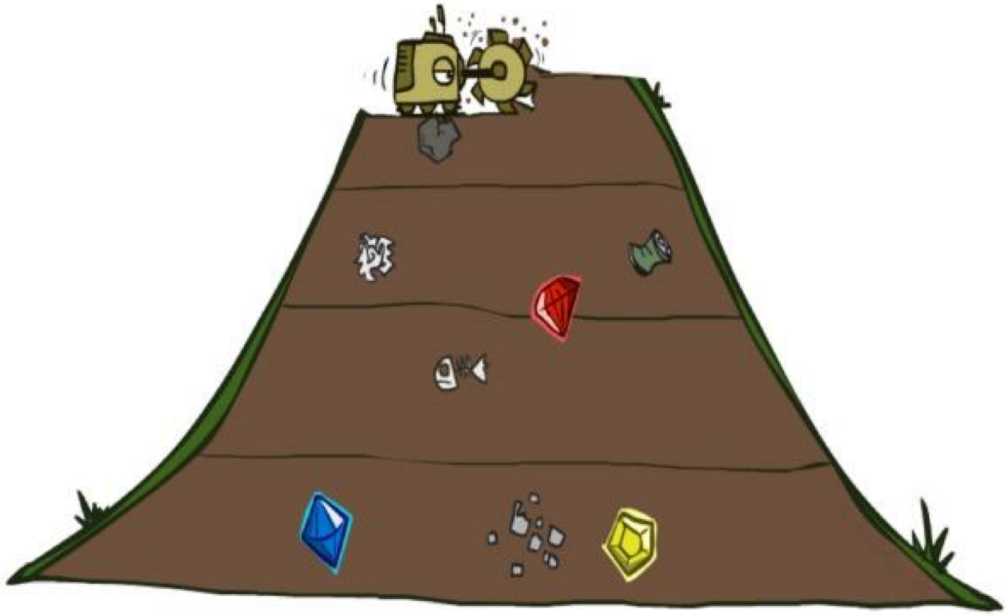
- **Uninformed strategies use only the information available in the problem definition**
  - Depth-first search
  - Breadth-first search
  - Uniform-cost search
  - Depth-limited search
  - Iterative deepening search

Depth-First Search

- Strategy: Expand the deepest node first until going deeper is not possible. If the goal is not found, expand another node.
- Implementation: Fringe is LIFO (Last In First Out) stack

Breadth-First Search

- Strategy: expand a shallowest node first
- Implementation: Fringe is a FIFO (First in First Out) queue

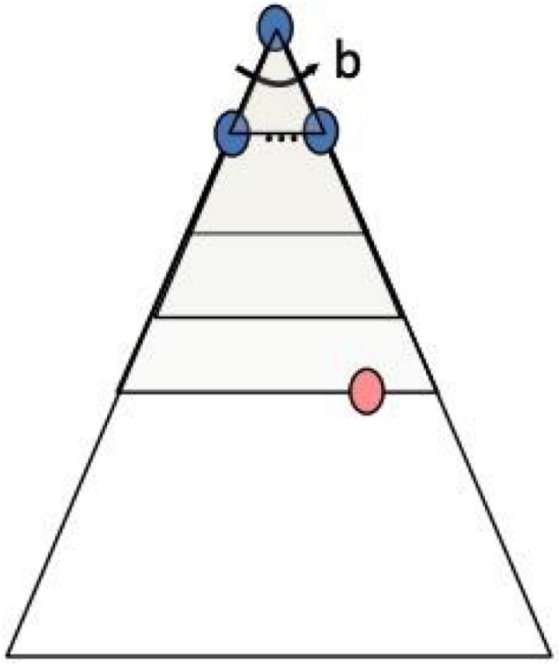


DFS vs. BFS

- When will DFS outperform BFS?
- When will BFS outperform DFS?

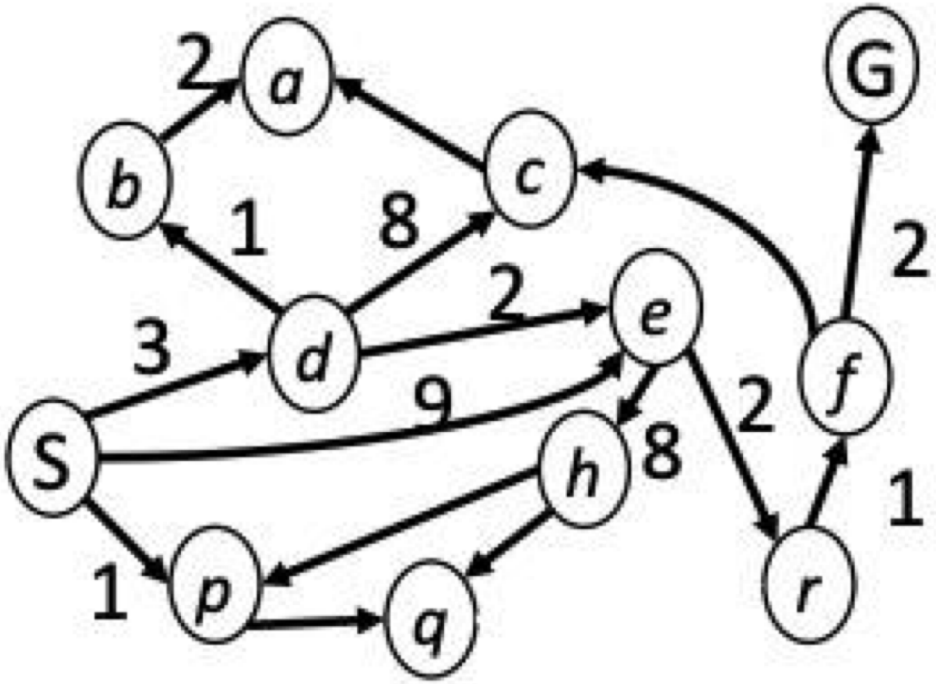
Iterative Deepening

- Idea: Get **DFSs space advantage** with **BFS's time / shallow-solution advantages**
  - Run a DFS with depth limit 1. If no solution…
  - Run a DFS with depth limit 2. If no solution…
  - Run a DFS with depth limit 3



Cost Sensitive Search

- BFS finds the shortest path in terms of number of actions.
- It does not find the least-cost path.



Uniform Cost Search

- Strategy: Expand the cheapest node first
- Fringe is a priority queue (priority: cumulative cost)

Uniform Cost Search

- UCS s **complete** and **optimal**
- Explores options in every direction
- No information about goal location

Summary

Criterion	Breadth-First	Uniform-Cost	Depth-First	Depth-Limited	Iterative Deepening
Complete?	Yes*	Yes*	No	Yes, if $l \geq d$	Yes
Time	$O(b^d)$	$O(b^{C^*/\epsilon})$	$O(b^m)$	$O(b^l)$	$O(b^d)$
Space	$O(b^d)$	$O(b^{C^*/\epsilon})$	$O(bm)$	$O(bl)$	$O(bd)$
Optimal?	Yes*	Yes	No	No	Yes*