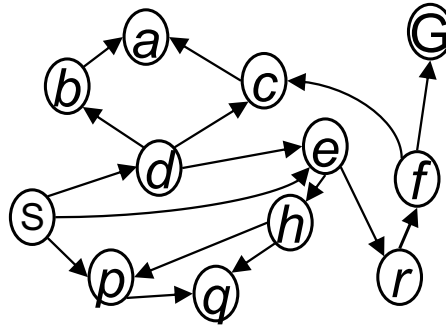# Breadth-First Search

Breadth-First Search

Strategy: expand a shallowest node first
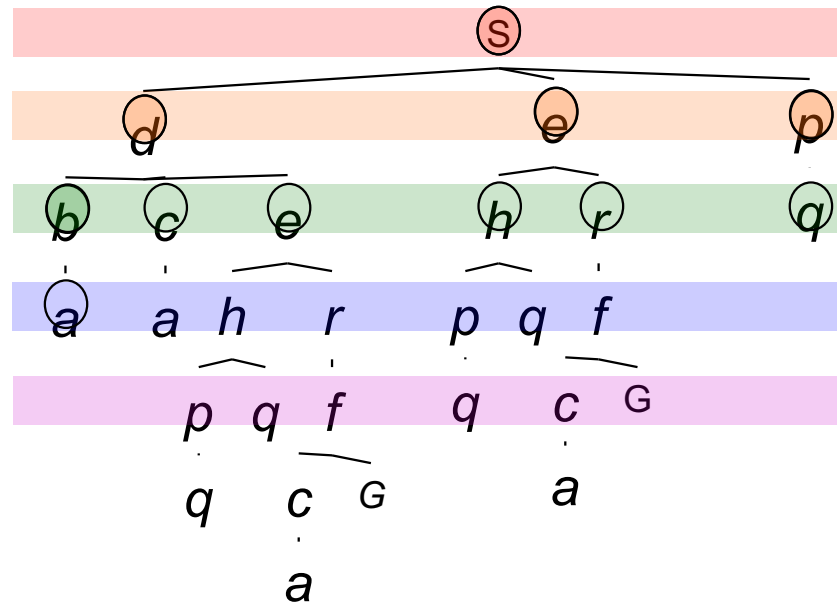
Implementation: Fringe is a FIFO queue

Search

Tiers

Breadth-First Search (BFS) Properties

- **What nodes does BFS expand?**
    - Processes all nodes above shallowest solution
    - Let depth of shallowest solution be s
    - Search takes time $O(b^s)$

- **How much space does the fringe take?**
    - Has roughly the last tier, so $O(b^s)$

- **Is it complete?**
    - s must be finite if a solution exists, so yes!

- **Is it optimal?**
    - Only if costs are all 1 (more on costs later)

s tiers

b

1 node
b nodes
b^2 nodes

b^s
nodes

b^m
nodes
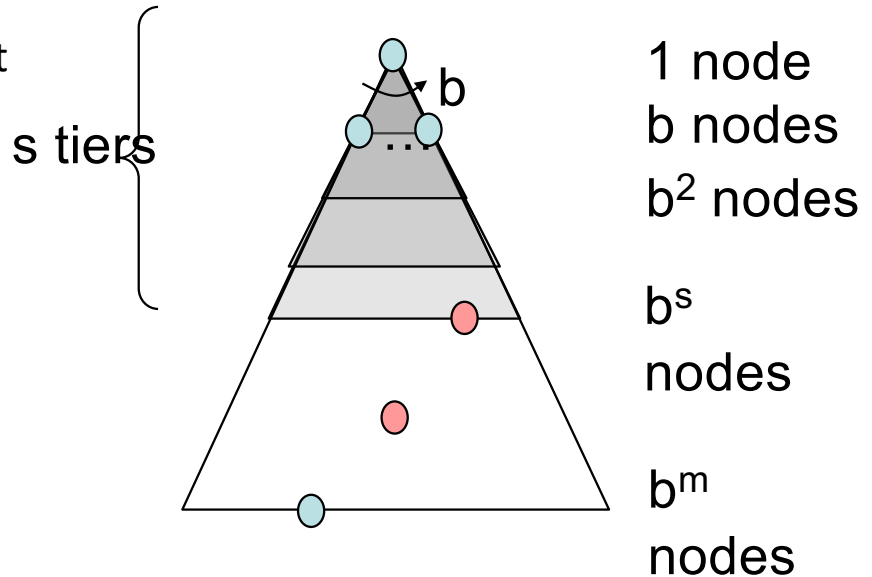
Quiz: DFS vs BFS

Quiz: DFS vs BFS

- **When will BFS outperform DFS?**

- **When will DFS outperform BFS?**

[Demo: dfs/bfs maze

Video of Demo Maze Water DFS/BFS (part 1)

Video of Demo Maze Water DFS/BFS (part 2)
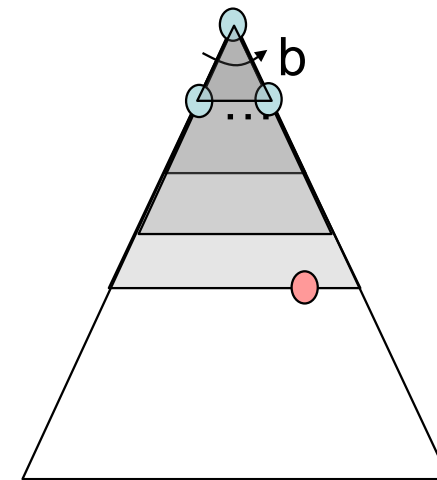
Iterative Deepening
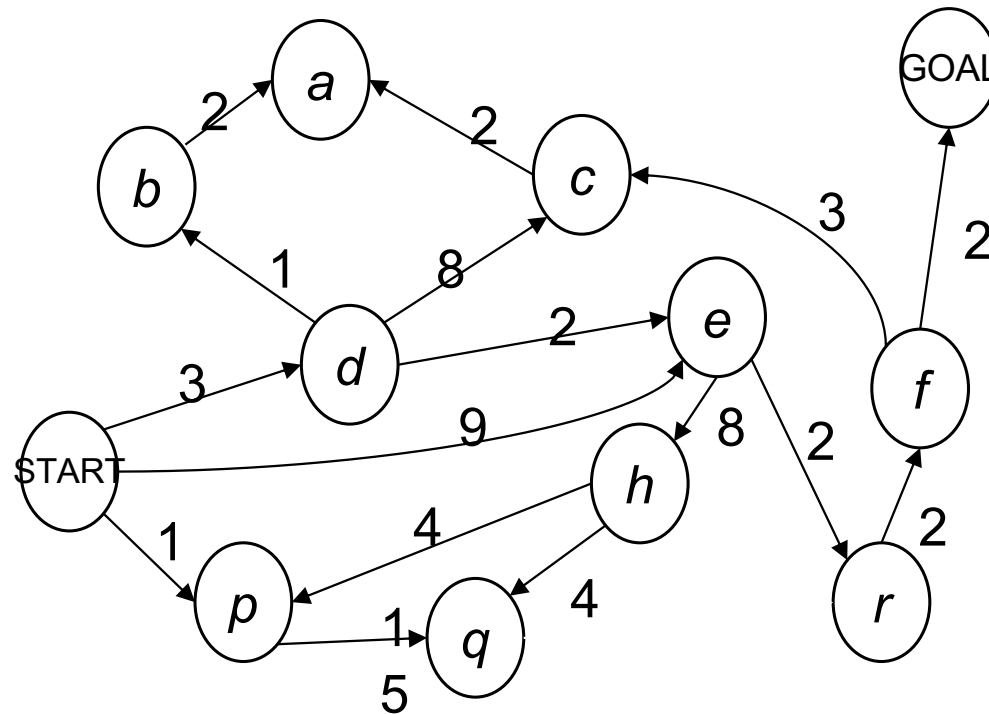
- Idea: get DFS's space advantage with BFS's time / shallow-solution advantages

  - 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. …..

- Isn't that wastefully redundant?

  - Generally most work happens in the lowest level searched, so not so bad!

Cost-Sensitive Search

BFS finds the shortest path in terms of number of actions.
It does not find the least-cost path. We will now cover
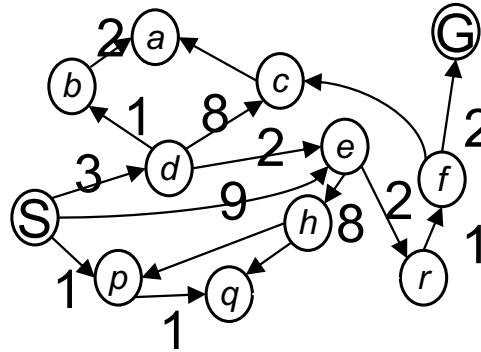a similar algorithm which does find the least-cost path.
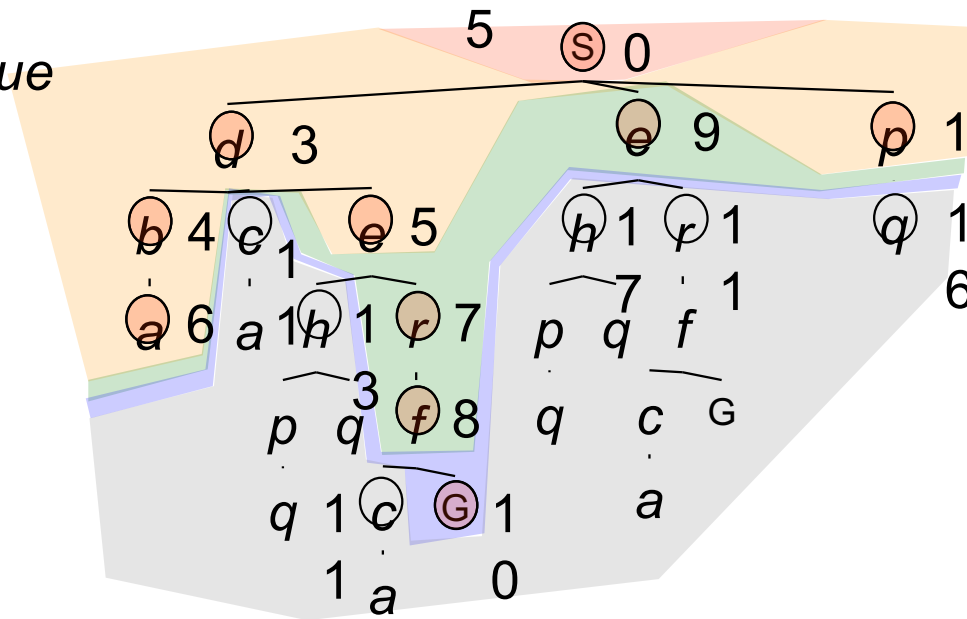
Uniform Cost Search

Uniform Cost Search

Strategy: expand a cheapest node first:

Fringe is a priority queue (priority: cumulative cost)

Cost contours

Uniform Cost Search (UCS) Properties

- **What nodes does UCS expand?**

  - Processes all nodes with cost less than cheapest solution!

  - If that solution costs C^* and arcs cost at least ε, then the "effective depth" is roughly C^*/ε

  - Takes time $O(b^{C^*/\varepsilon})$ (exponential in effective depth)
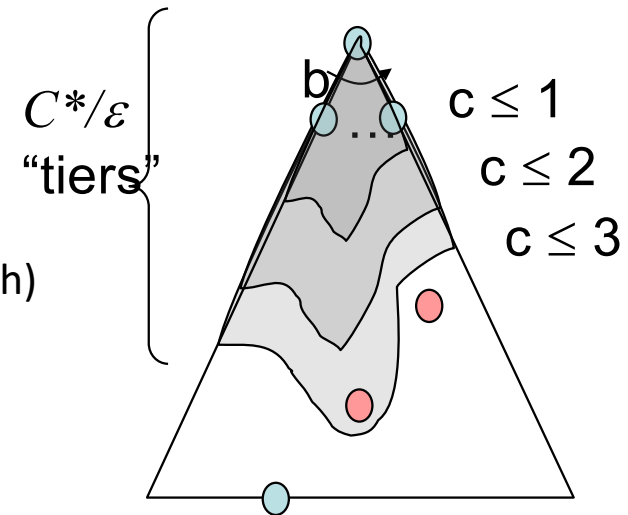
- **How much space does the fringe take?**

  - Has roughly the last tier, so $O(b^{C^*/\varepsilon})$

- **Is it complete?**

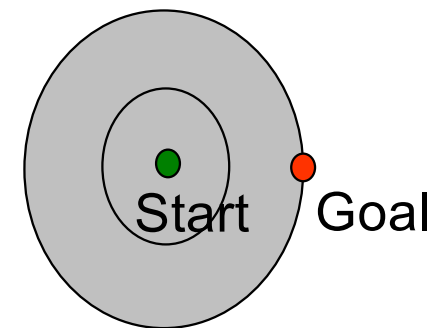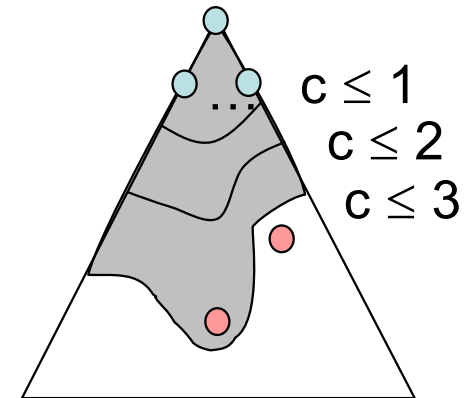 - Assuming best solution has a finite cost and minimum arc cost is positive, yes!

- **Is it optimal?**

  - Yes! (Proof via A*)

C^*/ε

"tiers"

b

$c \leq 1$

$c \leq 2$

$c \leq 3$

Uniform Cost Issues

- 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!

$c \leq 1$
$c \leq 2$
$c \leq 3$

Start   Goal

[Demo: empty grid UCS (L2D5)]
[Demo: maze with deep/shallow

Video of Demo Empty UCS

Video of Demo Maze with Deep/Shallow Water --- DFS, BFS, or UCS? (part 1)

Video of Demo Maze with Deep/Shallow Water --- DFS, BFS, or UCS? (part 2)

# Video of Demo Maze with Deep/Shallow Water --- DFS, BFS, or UCS? (part 3)

The One Queue

- All these search algorithms are the same except for fringe strategies
  - Conceptually, all fringes are priority queues (i.e. collections nodes with attached priorities)
  - Practically, for DFS and BFS, you can avoid the log(n) overhead from an actual priority queue, using stacks and queues
  - Can even code one implementation that takes a variable queuing object

Search and Models

- **Search operates over models of the world**
  - The agent doesn't actually try all the plans out in the real world!
  - Planning is all "in simulation"
  - Your search is only as good as your models...
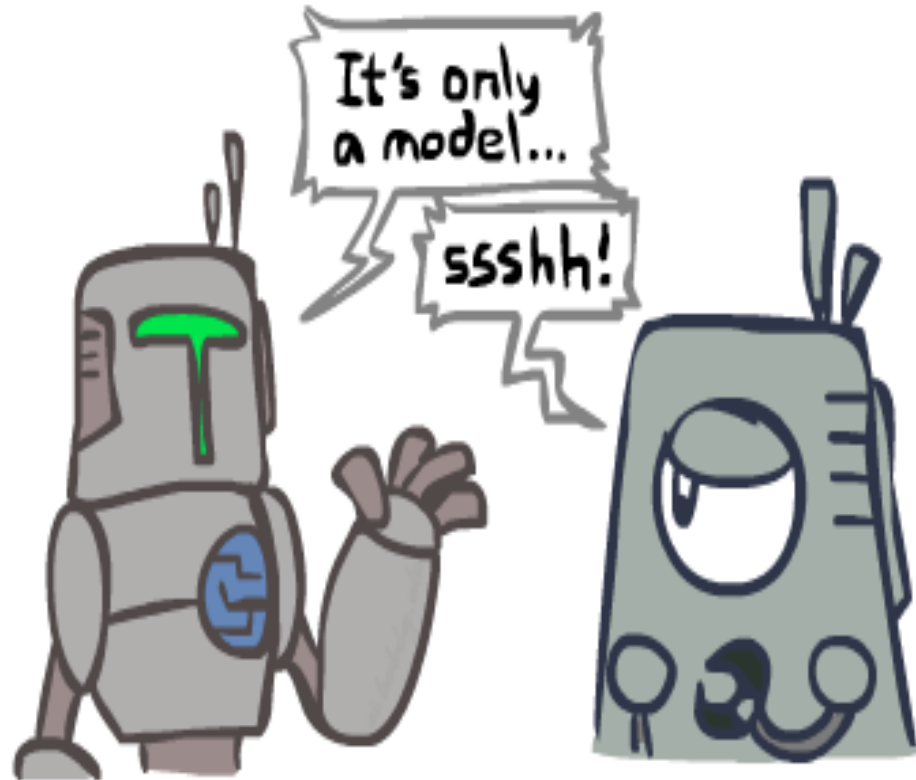
Search

Prof. Abbeel steps through a few search examples