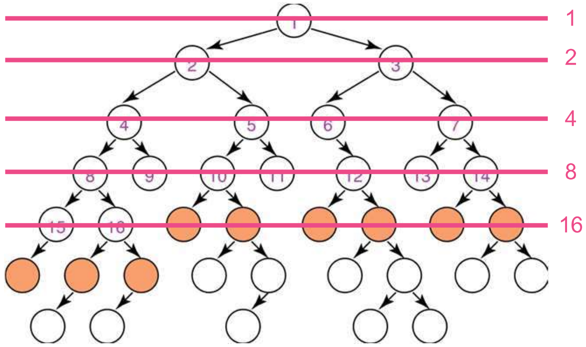
Properties of breadth-first search

Complete? Yes (if breadth, b, is finite, the shallowest goal is at a fixed depth, d, and will be found before any deeper nodes are generated)



Time? $1 + b^1 + b^2 + b^3 + \ldots + b^d = \frac{b^{d+1}-1}{b-1} = O(b^d)$

Space? $O(b^d)$) keeps every node in memory; generate all nodes up to level d)

Optimal? Yes, but only if all actions have the same cost

Space is the big problem for BFS. It grows exponentially with depth

Tree-Search DFS vs. Graph-Search DFS

Tree-Search DFS

- We covered Tree-Search DFS in this course
- Can be used for iterative deepening dfs
- Checks new states against those on the **path** from the **root** to the **current node** to avoid infinite loops in finite state space.
- Does not avoid the proliferation of redundant paths.
- Both are non-optimal
- The space complexity can be O(bm)

Graph-Search DFS

- You probably covered Graph-Search in COMP2521
- Can **not** be used for iterative deepening dfs!
- Keeps a record of every visited, expanded node
- Both are non-optimal
- The space complexity is as bad as BFS: O(b^m)

Properties of Iterative Deepening Search

Complete? Yes.

Time: $O(b^d)$

Space? $O(bd)$

Optimal? Yes, if step costs are identical.

In general, iterative deepening is the preferred search strategy for a large search space where depth of solution is not known

Properties of Uniform-Cost Search

Complete? Yes, if b is finite and if transition $cost \geq \epsilon$ with $\epsilon > 0$

Time? Worst case, $O(b^{\lceil C^*/\epsilon \rceil})$ where C^* = cost of the optimal solution
every transition costs at least ϵ
$\therefore$ cost per step is $\frac{C^*}{\epsilon}$

Space? $O(b^{\lceil C^*/\epsilon \rceil})$, $b^{\lceil C^*/\epsilon \rceil} = b^d$ if all step costs are equal

Optimal? Yes – nodes expanded in increasing order of lower path cost, $g(n)$

Complexity Results for Uninformed Search

Criterion	Breadth-First	Uniform-Cost	Depth-First	Depth-Limited	Iterative Deepening
Time	$O(b^d)$	$O(b^{\lceil C^*/\varepsilon \rceil})$	$O(b^m)$	$O(b^k)$	$O(b^d)$
Space	$O(b^d)$	$O(b^{\lceil C^*/\varepsilon \rceil})$	$O(bm)$	$O(bk)$	$O(bd)$
Complete?	Yes[1]	Yes[2]	No	No[4]	Yes[1]
Optimal ?	Yes[3]	Yes	No	No	Yes[3]

b = branching factor, d = depth of the shallowest solution,
m = maximum depth of the search tree, k = depth limit.

1 = complete if b is finite.

2 = complete if b is finite and step costs $\geq \varepsilon$ with $\varepsilon > 0$.

3 = optimal if actions all have the same cost.

4 = incomplete if the goal is not within the depth bound

UNSW SYDNEY

Properties of Greedy Best-First Search

Complete: No. Can get stuck in loops.
(Complete in finite space with repeated-state checking)

Time: $O(b^m)$, where m is the maximum depth in search space.

Space: $O(b^m)$ (retains all nodes in memory)

Optimal: No.

Greedy Search is not efficient

However, a good heuristic can reduce time and memory costs substantially.

Optimality of A*

- Heuristic h is said to be admissible if

  $\forall n h(n) \leq h^*(n)$ where $h^*(n)$ is the true cost from n to goal

- If h is admissible then $f(n)$ never overestimates the actual cost of the best solution through $n.$

  $f(n) = g(n) + h(n)$, where $g(n)$ is the actual cost to n and $h(n)$ is an underestimate

- Example: h = *straight line distance* is admissible because the shortest path between any two points is a line.
- A^* is optimal if h is admissible.
- Admissible heuristics are by nature optimistic because they think the cost of solving the problem is less than it actually is.

Optimality of A* Search

Complete: Yes, unless infinitely many nodes with $f \leq \textit{cost of solution}$

Time: Exponential in $\textit{relative error in } h \times \textit{length of solution}$

Space: Keeps all expanded nodes is memory

Optimal: Yes (assuming h is admissible).