TERMINOLOGY:

- A **node** in a graph represents state, parent, action, and path-cost.
- **Expanding** a node involved **generating** children & a goal test.
- The **fringe/frontier** is the set of reachable nodes yet to be expanded.
- We define a **strategy** as the criteria for choosing the next node to expand.

To decide on a search strategy, we need some criteria to evaluate them.
We have four major criteria:

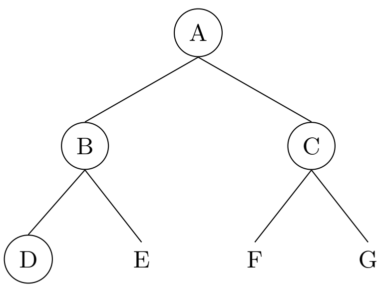
1. **completeness**: does it find a solution (if one exists)?
2. **optimality**: does it always find the best solution?
3. **time complexity**: number of worst case nodes expanded
4. **space complexity**: maximum number of nodes stored in memory

What do we use to measure these?

- b – max branching factor
- d – depth of least cost solution
- m – max depth of trees

Let's consider our two basic search algorithms:

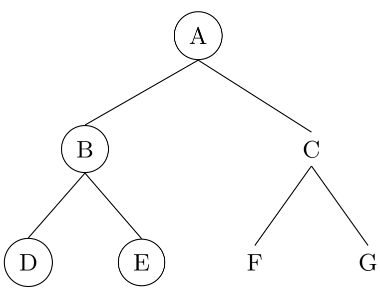
Breadth-First Search



Queue:
A
BC
CDE
DEFG
EFGHI...

1. Complete? Yes
2. Optimal? Yes
3. Time? $O(b^d)$
4. Space? $O(b^d)$

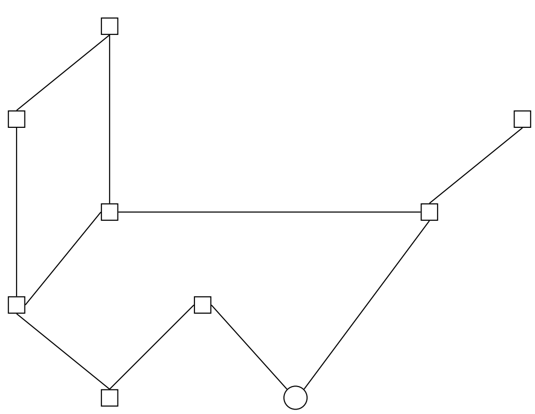
Depth-First Search



Stack:
A
CB
CED
CE...

1. Complete? No
2. Optimal? No
3. Time? $O(b^d)$
4. Space? $O(bm)$

We can't seem to improve the time complexity, but can we improve by blending BFS & DFS?
Consider a navigation problem:



We can see there are only 9 locations, so we can limit our path length!
We can use this to define a new algorithm:

Limited-Depth Search

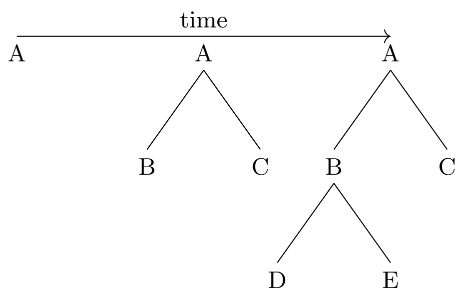
w/ length cutoff L, using priority queue (early = high priority)

1. complete? No
2. optimal? No
3. time? O(b^L)
4. space? O(bL)

We can extend this iteratively, increasing the length each round, for:

Iterative Deepening

where L denotes the iterative length.



1. Complete? No
2. Optimal? No
3. Time? $O(b^l)$
4. Space? $O(bl)$

But shouldn't we expect the time complexity to increase from limited-depth search?

$$t = b^0\left(\frac{b}{b-1}\right) + b^1\left(\frac{b}{b-1}\right) + ... + b^d\left(\frac{b}{b-1}\right)$$

$$\begin{aligned} \text{Constant} &= (b = 2) \rightarrow 2 \\ &= (b = 3) \rightarrow 1.5 \\ &= (b = 10) \rightarrow 1.1 \end{aligned}$$

Therefore the constant approaches 1!