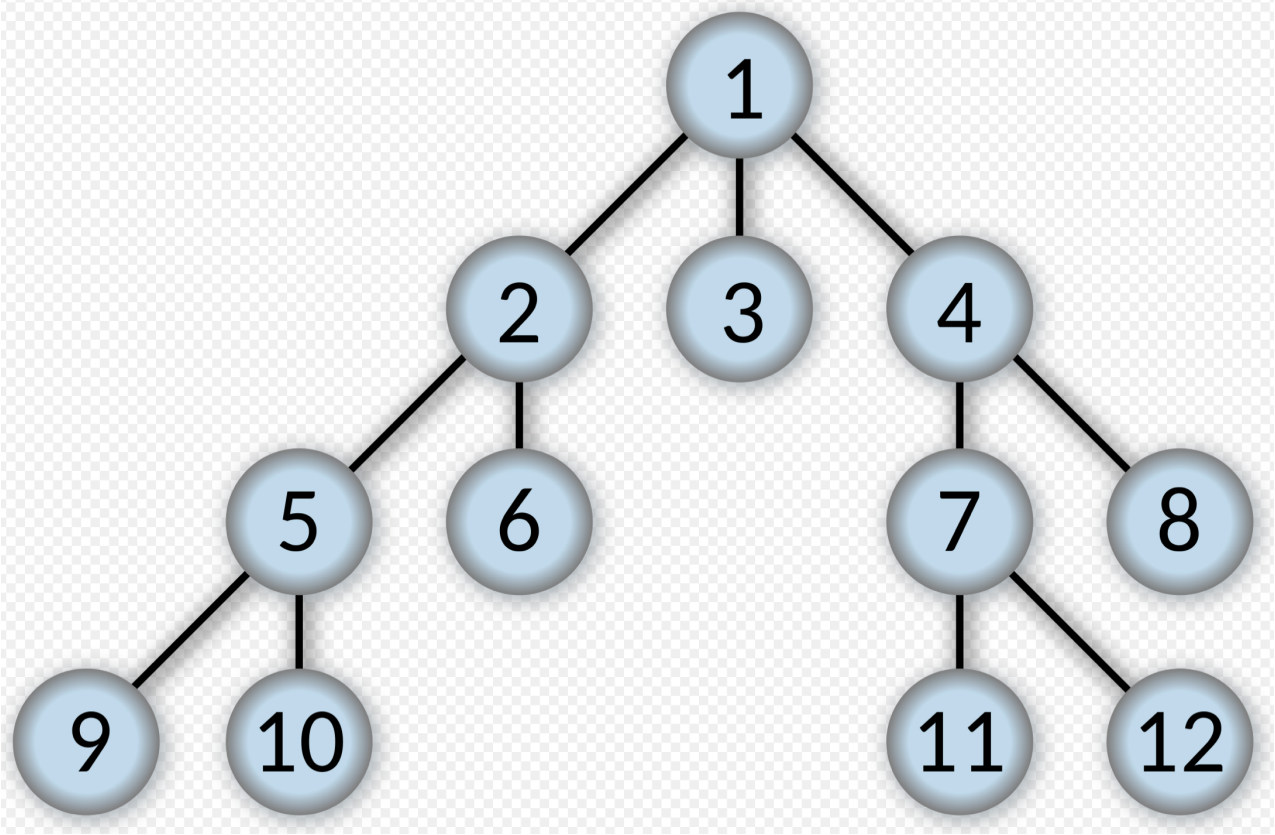
CSCI 3202: Intro to Artificial Intelligence

Lecture 4: Breadth–First Search (BFS)

Rachel Cox

Department of Computer Science



Search

Uninformed Search - no additional information about states beyond that in the problem definition

Informed Search - Some idea of which non-goal states are “more promising” than others

Search

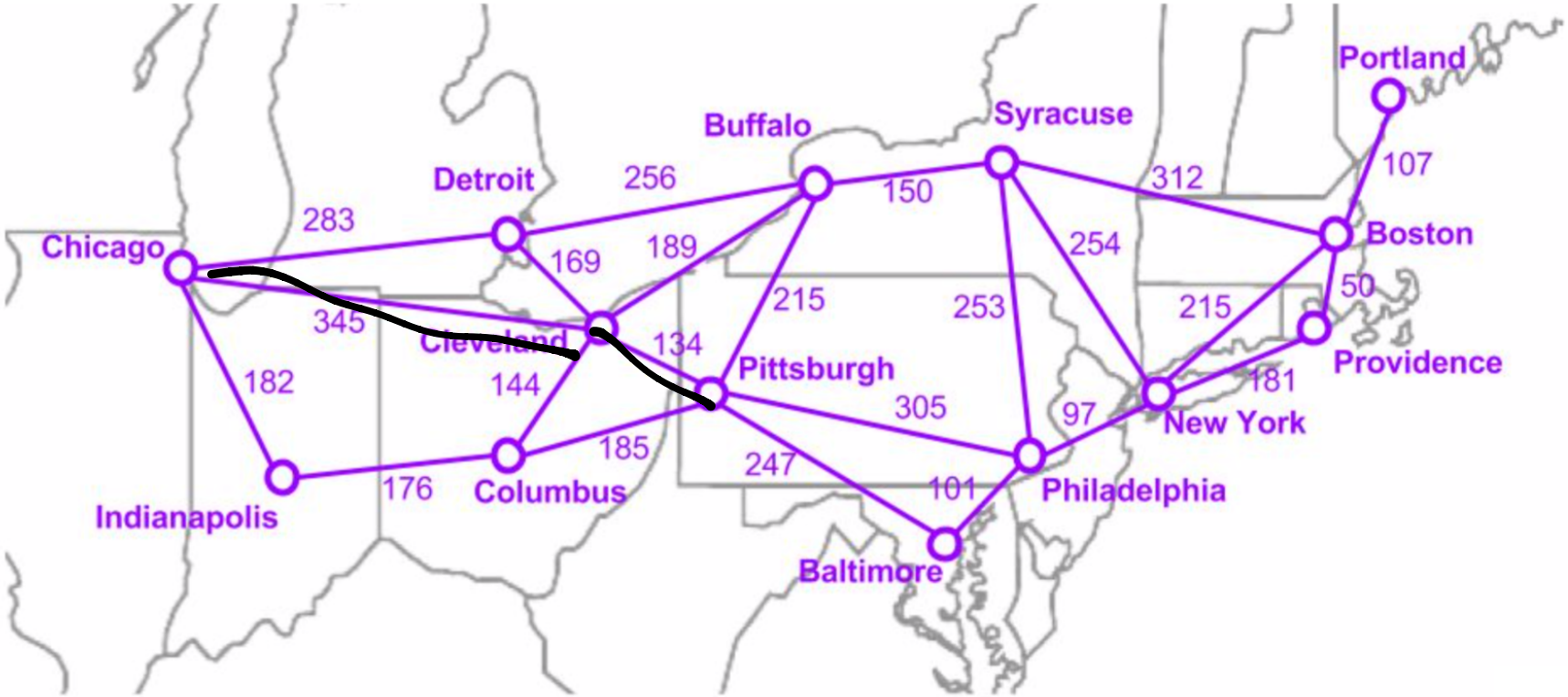
A few variables:

Branching factor: *b*, maximum number of successors of any node

Depth: *d*, the depth (in the search tree) of the shallowest goal node

$b = 5$

$d = 2$



Search

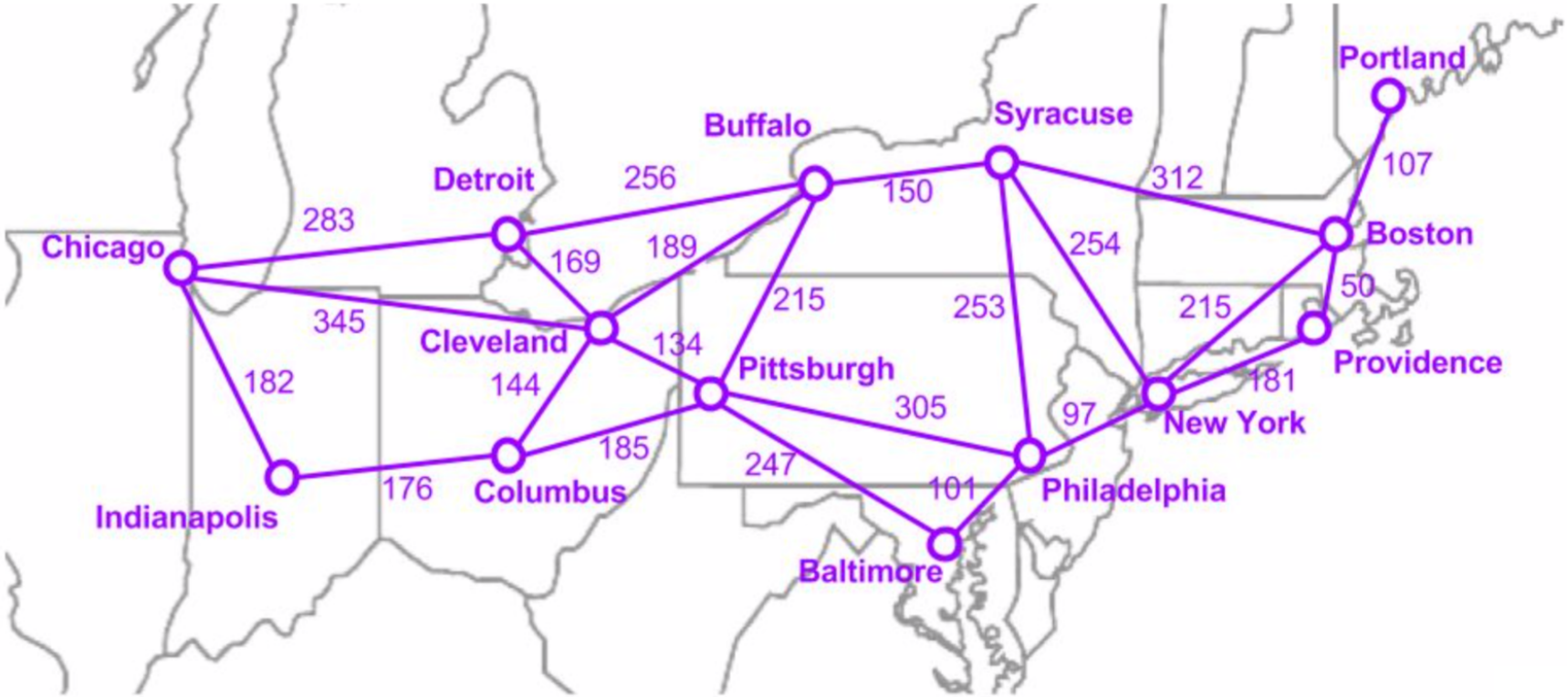
Things to think about:

Completeness: Is the algorithm guaranteed to find a solution, when one exists?

Optimality: Is the algorithm guaranteed to find the optimal solution (i.e. lowest path cost)

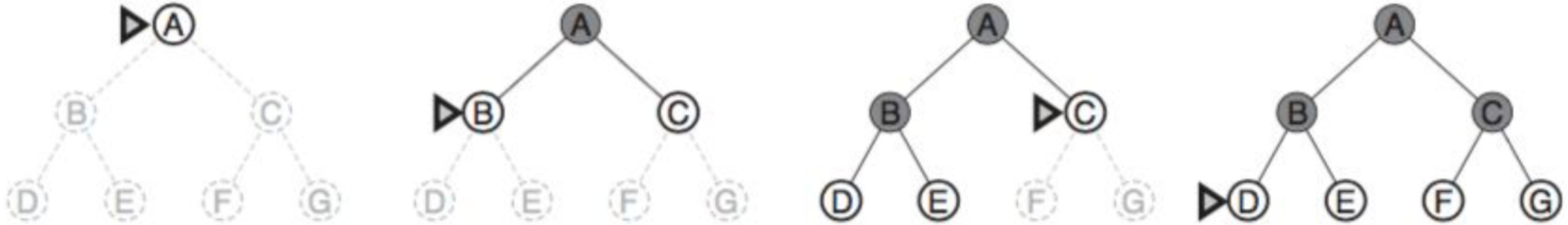
Time Complexity: How long does it take to find a solution?

Space Complexity: How much memory is needed to find the solution?



Breadth-first Search (BFS)

- Uninformed
- Expand all nodes at a given depth before proceeding into to the next layer (FIFO)
- Apply a goal test to each node ***as it is generated***



Explored:	{ }	{A}	{A, B}	{A, B, C}
Frontier:	[ ]	[B, C]	[C, D, E]	[D, E, F, G]

Breadth-first Search (BFS)

Example: Traveling in the US northeast

Start: Chi

goal: Pitt

Ex: {Chicago}

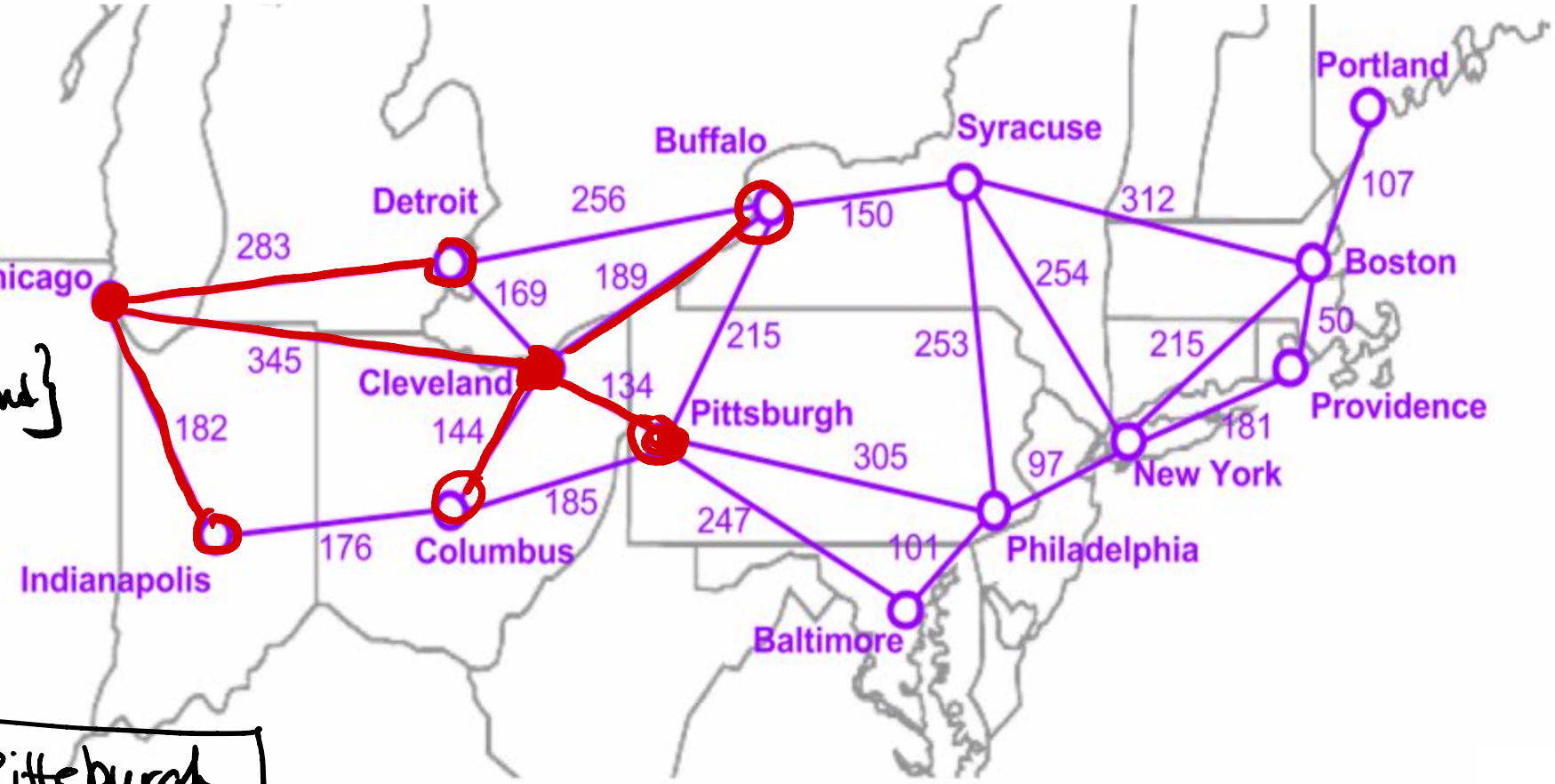
Front: {Cle, Det, Ind}

step 2: Exp: {Chicago, Cleveland}

Front: {Det, Ind, Buffalo, Columbus, Pittsburgh}

Chicago → Cleveland → Pittsburgh

Step costs: miles between cities along major highways



Breadth-first Search (BFS)

Example: Traveling in the US northeast

Question: Would changing the step cost function change our BFS result?
-Even if there is a major traffic jam in our route?

No! we are ignoring "path cost"

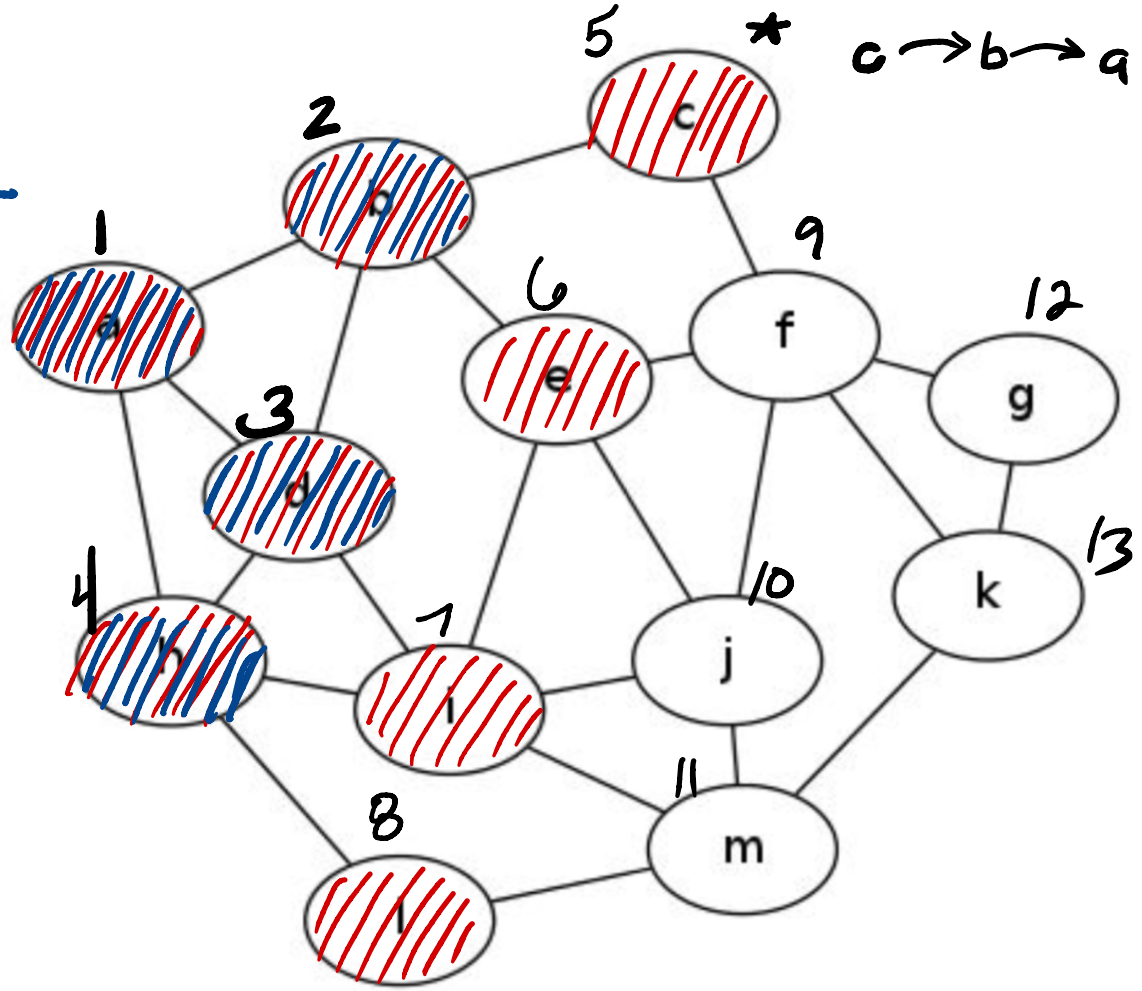
Step costs: estimated travel time (in minutes) along major highways

Breadth-first Search (BFS)

Frontier Explored

Example: Number the nodes in the search graph according to the order in which they would be expanded using BFS to find a path from a to k. Assume that nodes within a layer are expanded in alphabetical order.

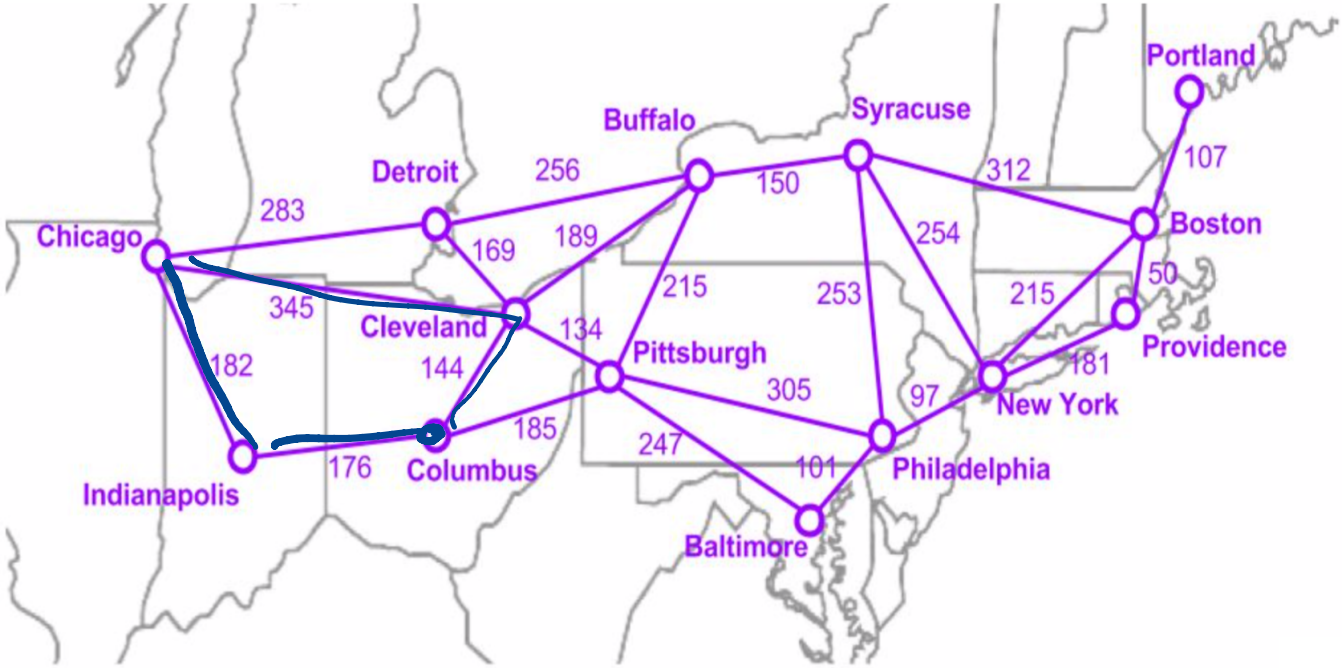
	Explored	Frontier
layer 0	{ }	[a]
layer 1	{a}	[b,d,h]
layer 2	{a,b}	[d,h,c,e]
	{a,b,d}	[h,c,e,i]
	{a,b,d,h}	[c,e,i,l]

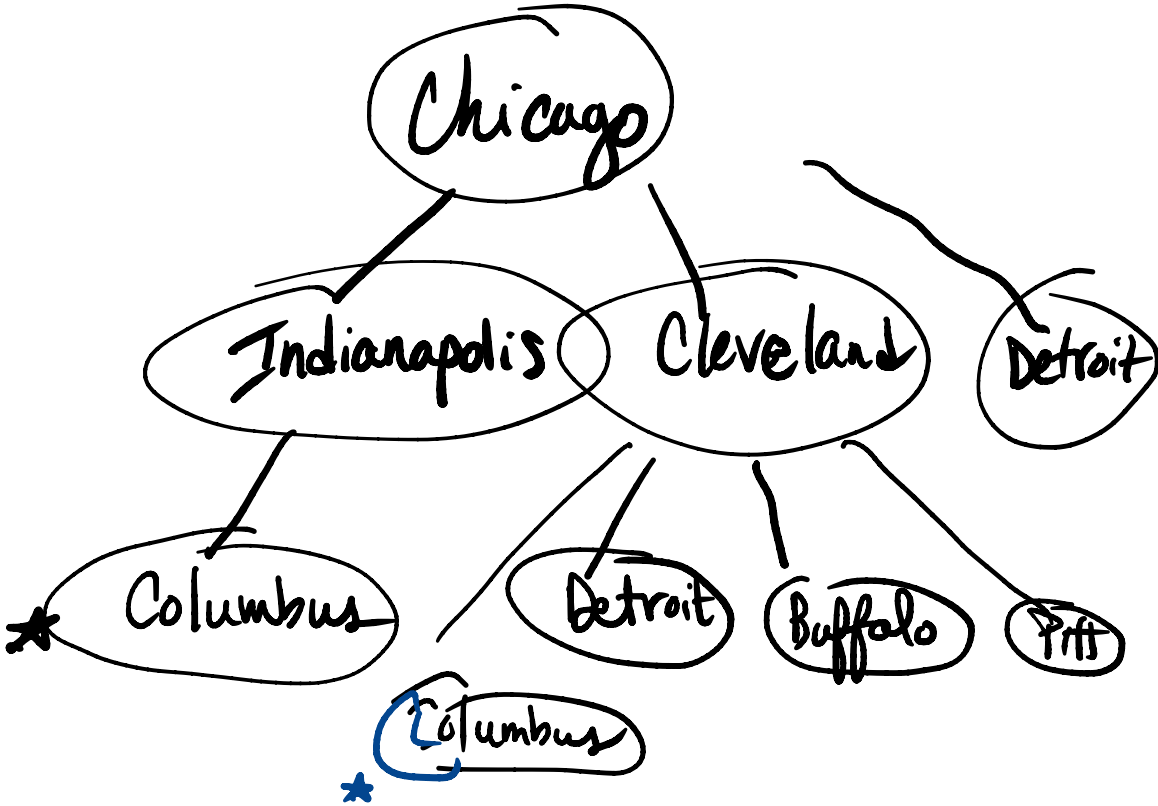


Breadth-first Search (BFS)

Example: Traveling in the northeast again. Sketch a search tree with Chicago as the initial state.

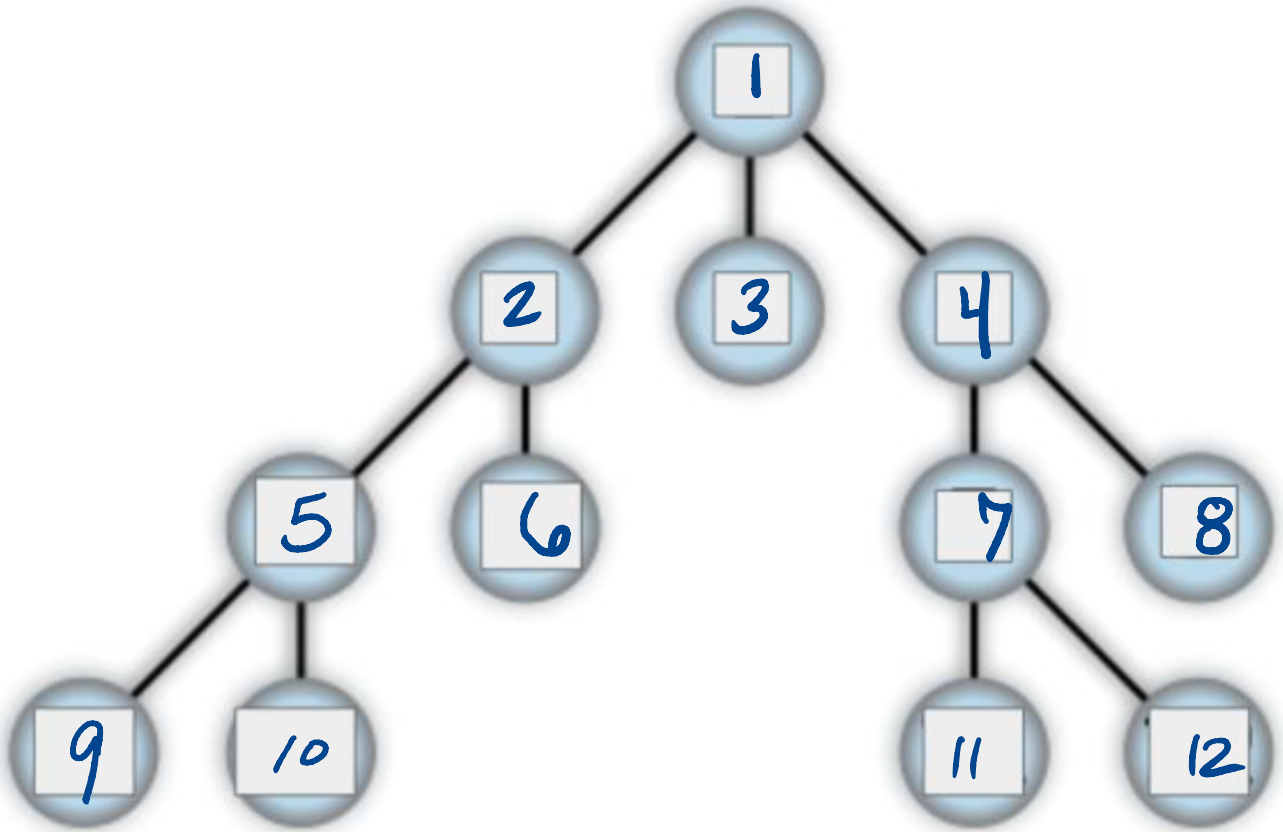
Chicago → Pittsburgh





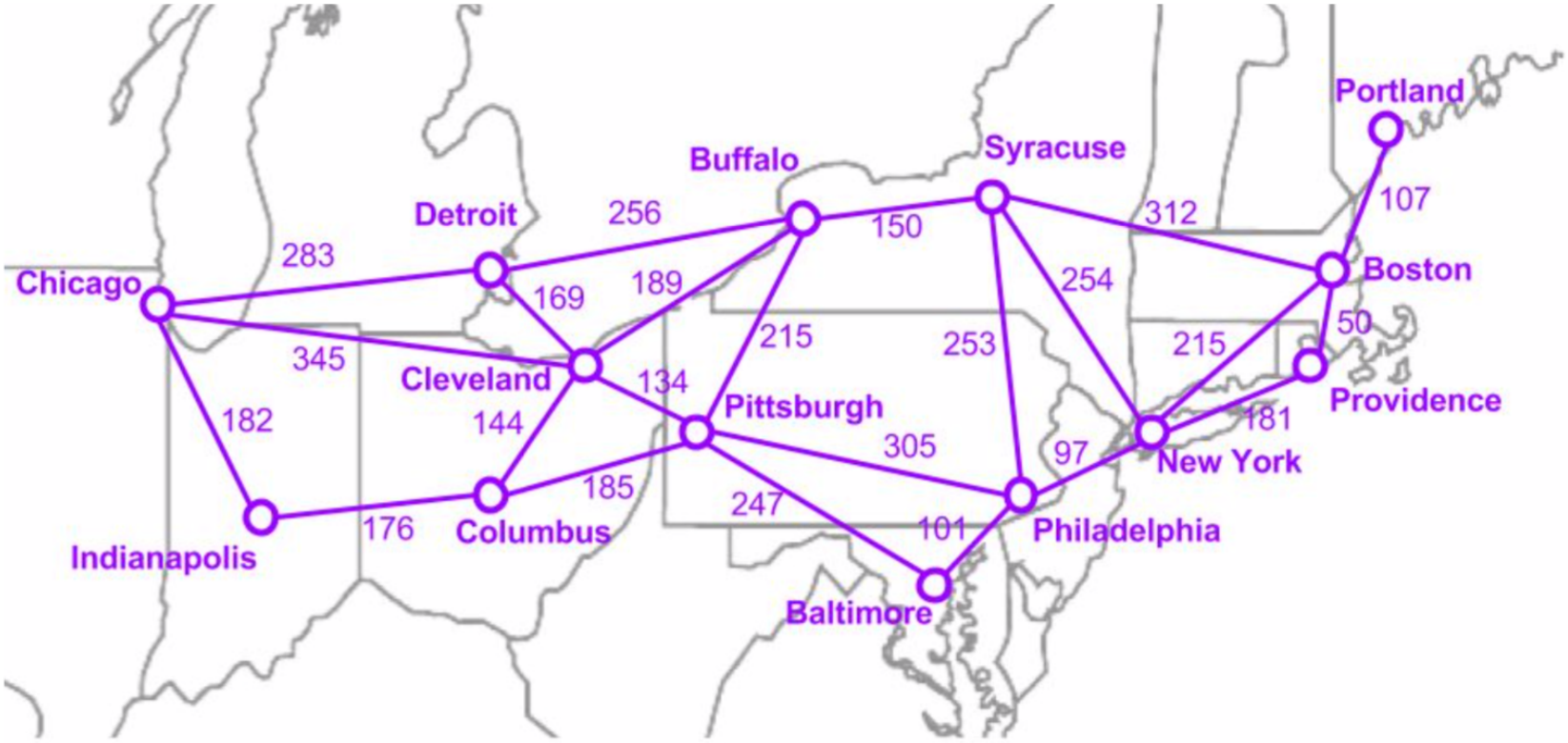
Breadth-first Search (BFS)

Example: Number the nodes in the search tree according to the order in which they would be expanded using BFS. Assume that the goal is not found, and nodes within a layer are expanded from left to right.



Breadth-first Search (BFS)

$b = 5$



Complete? yes! If a solution exists, will we find it. Definitly

Optimal? not guaranteed to be optimal

Breadth-first Search (BFS)

Worst case scenario

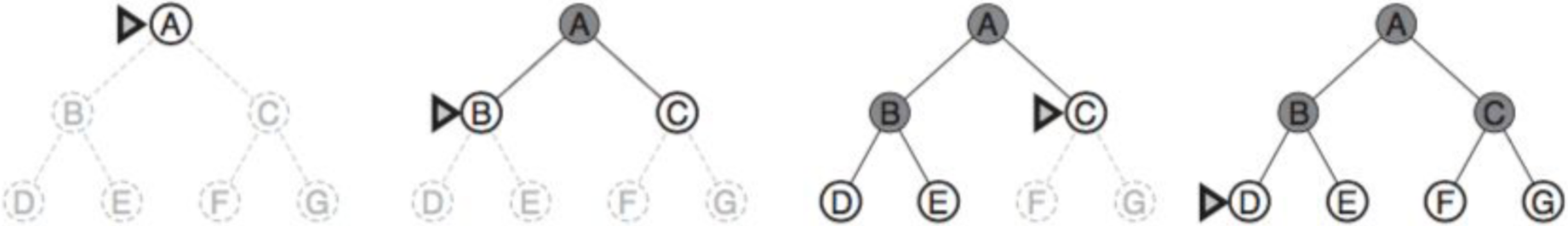
Time Complexity: Suppose that each layer generates b nodes (calling b the "branching factor") and the search problem has d total layers.

depth of your goal

- layer 0 (root) generates $b^0 = 1$ node
- layer 1 generates $b^1 = b$ nodes
- layer 2 generates b^2 nodes

… and so on …

total: $\mathbf{1 + b + b^2 + b^3 + \ \ldots + b^d = \mathcal{O}(b^d)}$

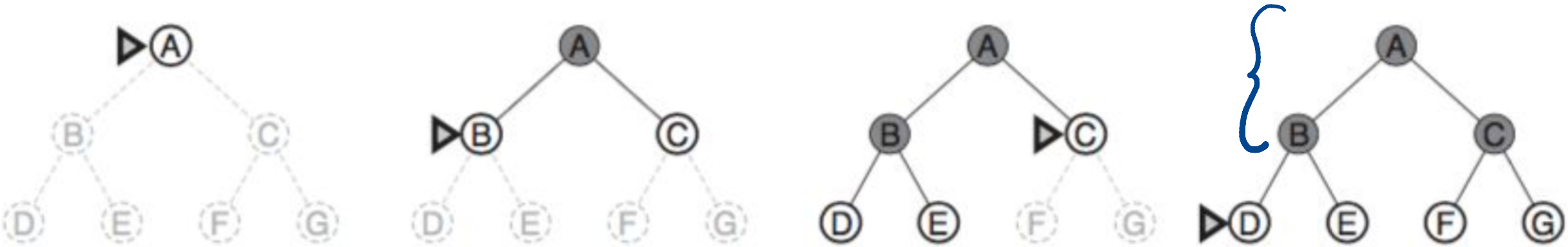


Breadth-first Search (BFS)

Space Complexity: need to store every node in the explored set $= \mathcal{O}(b^{d-1})$

and every node on the frontier $= \mathcal{O}(b^{d})$

- $\mathcal{O}(b^{d})$



Breadth-first Search (BFS)

Space issues are more significant than time issues

Memory requirements are a problem.

Depth	Nodes	Time	Memory
2	110	.11 milliseconds	107 kilobytes
4	11,110	11 milliseconds	10.6 megabytes
6	10^6	1.1 seconds	1 gigabyte
8	10^8	2 minutes	103 gigabytes
10	10^{10}	3 hours	10 terabytes
12	10^{12}	13 days	1 petabyte
14	10^{14}	3.5 years	99 petabytes
16	10^{16}	350 years	10 exabytes

Figure 3.13 Time and memory requirements for breadth-first search. The numbers shown assume branching factor $b = 10$; 1 million nodes/second; 1000 bytes/node.

Next Time

Depth-first Search (DFS)