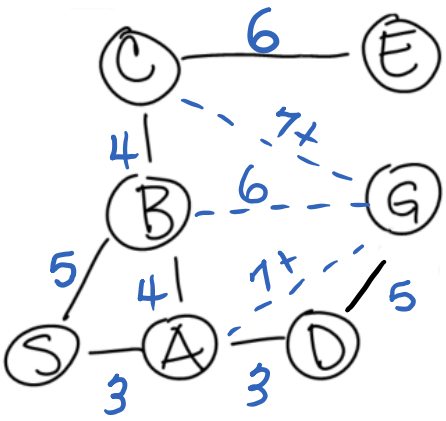
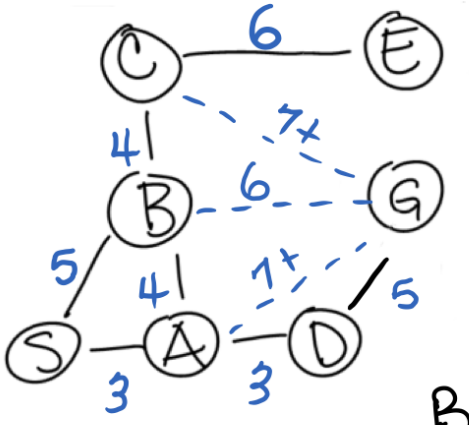


Search

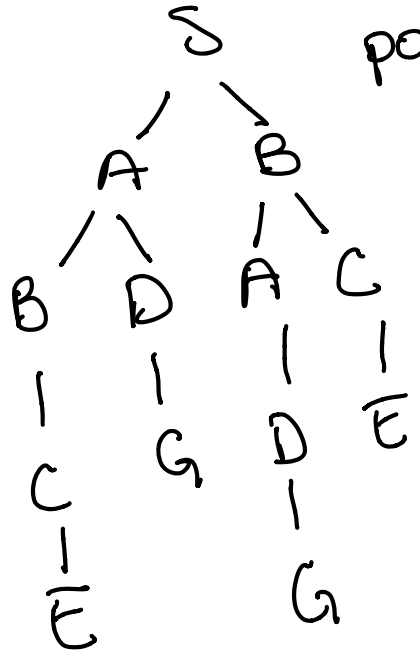
- i.e. finding shortest path
- ex: S to G

British Museum Algorithm

start @ S



Bri

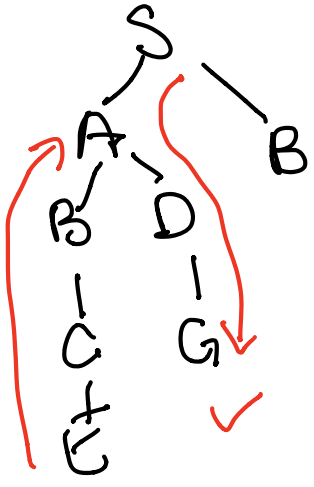


possible paths

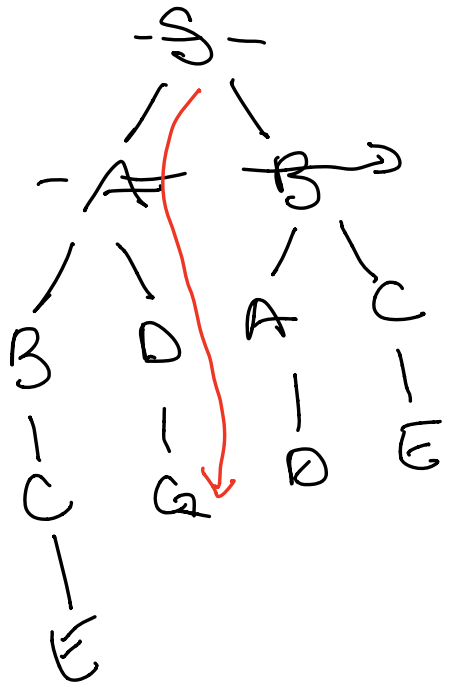
- searches every possible path

Depth First Search (start w/ left branch)

- can backtrack



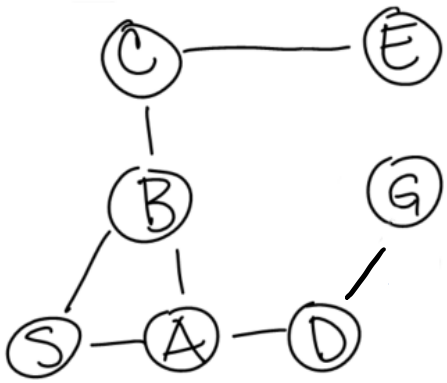
Breath First Search (BFS)



- takes a long time
- disadvantages:
  - went away from the goal
  - went to the same node more than once

- modify algorithm so it doesn't visit node more than once
  - use extended list

	Backtrack	extended list	informed	guarantee
British	X	X	X	✓
DFS	✓	✓	X	✓
BFS	X	✓	X	✓
HC	✓	✓	✓	✓
Beam	X	✓	✓	X



Initialize queue → check first path → extend first path if not already extended → enquire

DFS: front
BFS: back
HC: front, sort
Beam: anywhere keep w
Best: anywhere keep 1

if not @ goal, take it off

DFS: ~~(S)~~
(~~SA~~)(SB)
(~~SAB~~)(SAD)(SB)
(~~SABE~~)(SAD)(SB)
(~~SADG~~)(SB)
(SADG))(SB

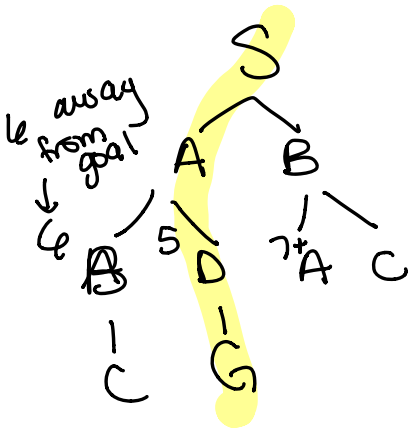
Hill Climbing (type of DFS)

- informed search
- might actually go to a local maxima

Beam search (only keep promising ways)

w=2 (keep 2 best paths)

- use heuristics
- length may be limited



Best first: (choose best node & extend out)

How to know which one to use?

- what kind of problem is it?
  - Size of soln space (i.e. have a lot of solns → dfs would take a long time)