Learning Objectives

- To practice running BFS, DFS, and UCS on a small graph
- To apply the search algorithms to a new, real-world problem.

Q1. Graph Search

Observe the graph below.

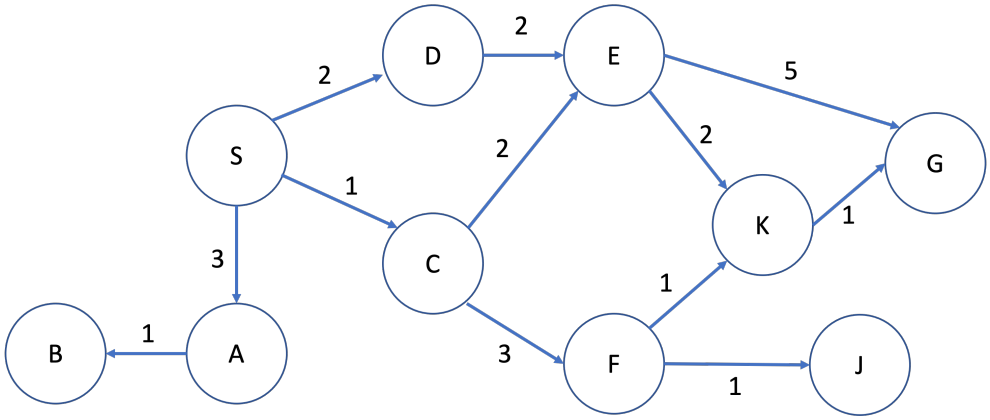


Figure 1: Graph

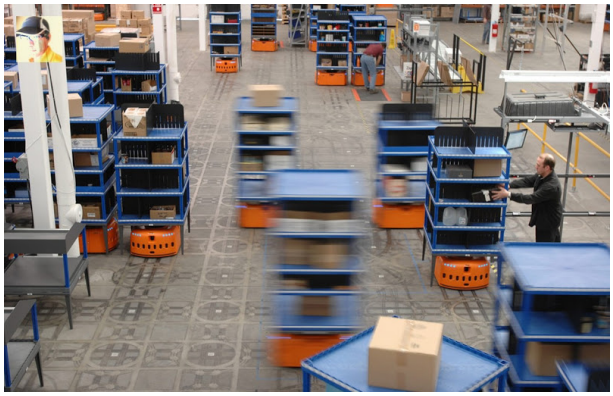
Write the sequence of states in the order they are **explored** when we search for paths from $S \rightarrow G$, break ties alphabetically. Then, write the path that is returned by each algorithm. As you work, think of which data structures are being used, the run-time complexities of each algorithm, and the shortest and optimal (lowest cost) paths.

BFS Explored Order: S, A, C, D, B, E, F, G	**BFS Path:** S, C, E, G	**BFS Path Cost:** 8
DFS Explored Order: S, A, B, C, E, G	**DFS Path:** S, C, E, G	**DFS Path Cost:** 8
UCS Explored Order: S, C, D, A, E, B, F, J, K, G	**UCS Path:** S, C, E, K, G	**UCS Path Cost:** 6

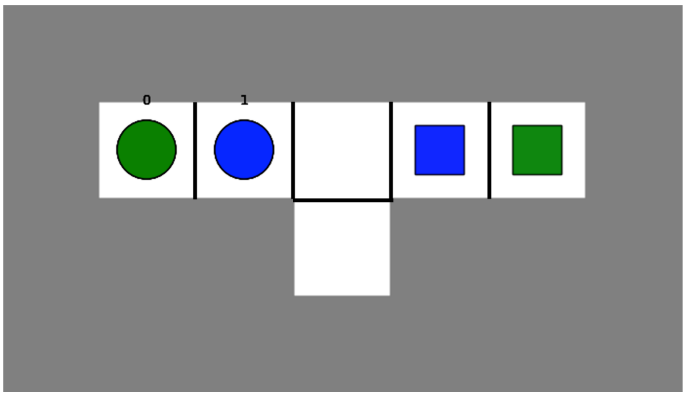
Space for scratch work.

Q2. Amazon Warehouse Robots

Amazon warehouse robots transport shelves of products autonomously to people who pack orders into boxes. The robot navigation allows people avoid wasting time walking through the facility to find items and instead to focus on the challenging task of picking up items of all shapes of sizes and putting them in appropriately-sized boxes. A picture of the robots is shown in Figure a.



(a) Amazon Warehouse Robots



(b) Small Warehouse Example

The robots navigate on a grid from one square to the next (left/right/up/down). Assume the Amazon robots are omnidirectional – they can move in any direction at any time (i.e., their orientation does not matter). Robots cannot occupy the same square at the same time, nor can they travel on the same edge between adjacent squares at the same time (even in opposite directions). Cameras on the ceiling keep track of all the robots as they navigate.

An example of a small 6-square warehouse with two robots - blue and green circles - is shown in Figure b along with their respective goals (same colored squares).

(a) If only one robot was in a warehouse of size M squares by N squares, what is the state space of the robot? If there were 15 robots in the same warehouse, how big would the state space be?

Single Agent State Space	**Multiple Agent State Space**
MN	$(MN)^{15}$

(b) Consider the small 6-square warehouse above with two robots.

(i) What single agent path planning algorithm would you choose for each robot? The state would only include one robot and would be ignoring the placement of the other robot.

● BFS ○ DFS ○ UCS

(ii) Use the algorithm to plan paths for each robot in Figure (b). Is there anything wrong with those paths? Why or why not? The paths collide when the blue robot stops before the green robot can pass by. In order to fix the problem, the blue robot needs to know that it should move out of the way to let the green robot go in front. The only way to do this is for the blue robot to know where the green robot is going and to check for collisions. We'll do it using joint planning.

(iii) A joint plan is one in which multiple agents decide together (jointly) what their plans should be. Let's try using BFS with the joint state space to find the joint plan. Each state now represents the tuple of both robots' locations, and each action planned represents both robots moving at the same time. Does BFS with the joint state space find a valid solution? What is the joint plan to solve this navigation problem? Yes, because BFS would know about collisions, it can avoid doing that and return a valid solution. The joint plan has both robots move forward, then the blue one moves down and the green moves forward, then after the green has passed, the blue comes up and both end at their goals.

(iv) Why might Amazon NOT want to find a joint plan by searching the joint state space in their real warehouses? They want to have lots of robots, but the state space grows exponentially.

(v) Can you think of a way to search quickly to find many robot paths but still ensure that the robots don't collide?