Learning Objectives

- To practice evaluating admissibility/consistency of heuristics and running Greedy and A* search on a graph.
- To compare different search heuristics and their performance on a real-world application.

Q1. Informed Search

Observe the graph below.

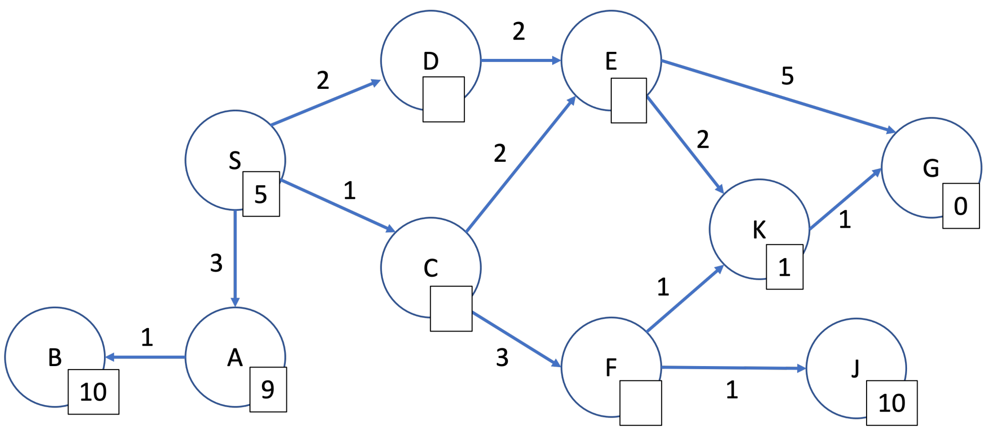


Figure 1: Graph

(a) Fill in the heuristic values such that the heuristic is admissible.

Is your heuristic also consistent? What would you need to change to make it consistent? We chose C:3, D:3, E:2, F:2. It is not consistent because S-C is 1 but 5-3=2. We would need to increase C to at least 4.

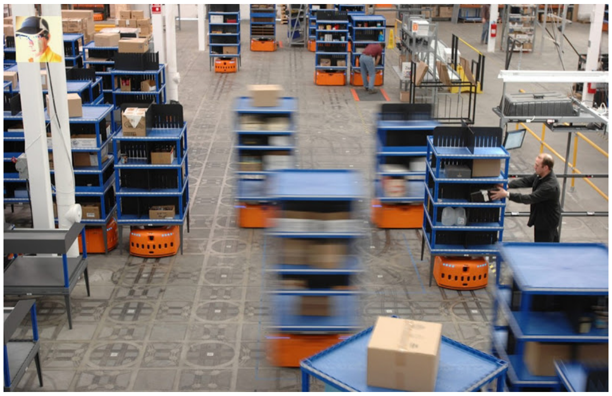
(b) Let's try to run graph search with your admissible heuristic. Run UCS, Greedy, and A* search from $S \to G$, break ties alphabetically. Write the order that the states are explored and the path that they find. As you work, think of which data structures are being used, the run-time complexities of each algorithm, and the shortest and optimal (lowest cost) paths.

UCS Explored Order: S, C, D, A, E, B, F, J, K, G (same as lecture 2)	**UCS Path:** S, C, E, K, G	**UCS Path Cost:** 6
Greedy Explored Order: S,C,E,G	**Greedy Path:** S, C, E, G	**Greedy Path Cost:** 8
A* Explored Order: S,C,E,D,K,G	**A* Path:** S, C, E, K, G	**A* Path Cost:** 6

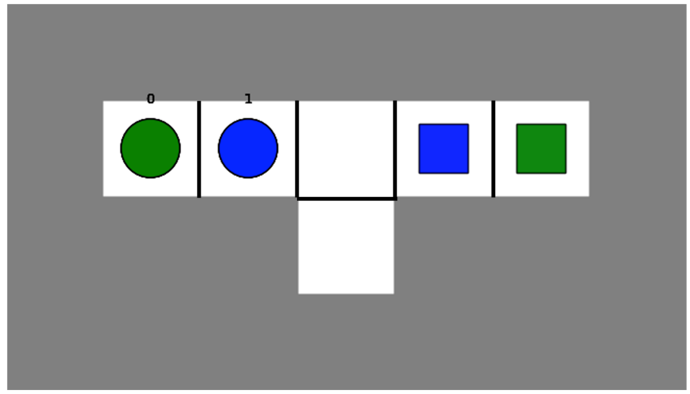
Space for scratch work.

Q2. Amazon Warehouse Robots

Recall the Amazon Warehouse Robots from last lecture.



(a) Amazon Warehouse Robots



(b) Small Warehouse Example

(a) What is a consistent heuristic that we could use for multi-agent A* search? One good heuristic is the max of the Manhattan distances for each robot to the goal. If the robots all moved at the same time, it would take the longest travel time to finish.

(b) Do you think that A* with this heuristic search on the joint state space will speed the search enough so that it could be usable by Amazon? Why or why not? It does not help significantly because in order to move around each other, the robots do not follow the Manhattan paths very often.

(c) The real heuristic algorithm that Amazon uses to find paths in its warehouses is the following. It randomly sorts the robots in some order. Then, it performs single agent search for the first robot. It passes the first robot's plan as additional constraints for where the second robot cannot move, and then solves for the second robot's plan. It keeps passing all previous robot plans as constraints for the next robot until all the robots have plans.

(i) Is this heuristic complete (always finds a valid solution to moving all the robots if one exists)? If yes, how do you know? If not, provide a counter example.

(ii) Is this heuristic optimal (always finds the best – shortest path – solution)?

It is not complete nor is it optimal. The shown example is a counterexample for completeness. If you plan the blue robot first, then the green one can never arrive at the goal. Order of the robots matters. However, it is so much faster than joint state space search even if it has to run several times with different robot orders.