CS444 Search Exercise

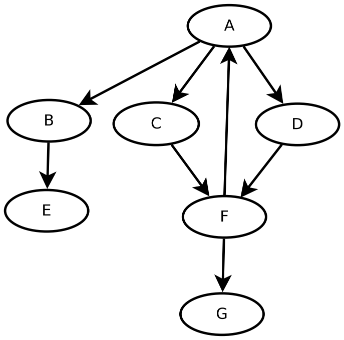
1. **DFS Exercise**

   - Open the AISpace search tool.
   - Load the “Extended Tree Graph” problem:
     `File` → `Load Sample Problem` → `Extended Tree Graph`
   - Open the “Solve” tab.
   - Select “Depth First Search” from the “Search Options” menu.
   - Step through the search using the “Fine Step” button.
     - Can you anticipate which node will be chosen from the frontier?
     - What is the largest number of nodes that end up on the frontier during this search?
     - In general, if the forward branching factor is b for all nodes, and the length of the search path is d how many nodes may end up on the frontier?

2. **BFS Exercise**

   - Repeat the previous steps using breadth-first search:
   - Step through the search using the “Fine Step” button.
     - Can you anticipate which node will be chosen from the frontier?
     - What is the largest number of nodes that end up on the frontier during this search?
     - In general, if the forward branching factor is b for all nodes, and the length of the search path is d how many nodes may end up on the frontier?
   - Which algorithm used more space for the frontier during the search?

The next three questions refer to the following state transition graph. For each question assume that A is the start state, E is the goal state, and each action has a cost of 1. Complete each search using multiple-path pruning.



3. **DFS by Hand**

   Fill in the following table by tracing the execution of a DFS search of the graph above. (When a node is expanded, the children should be added to the frontier in alphabetical order.)

Iteration #	Frontier	Explored
1.	A	-
2.		
3.		
4.		
5.		
6.		
7.		
8.		

4. **BFS by Hand**

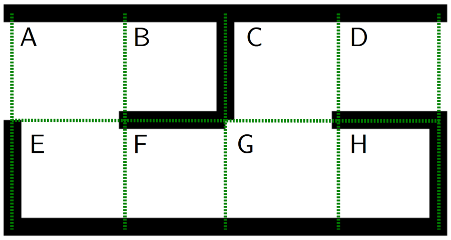
   Fill in the following table by tracing the execution of a BFS search of the graph above. (When a node is expanded, the children should be added to the frontier in alphabetical order.)

Iteration #	Frontier	Explored
1.	A	-
2.		
3.		
4.		
5.		
6.		
7.		
8.		

5. It turns out that both searches considered above find the optimal solution to this problem. Modify the graph to create a new problem that causes DFS to find a non-optimal solution. Will BFS find the optimal solution for this new problem?

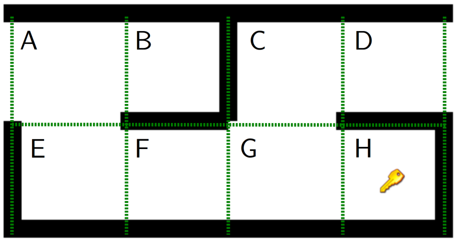
6. **Dungeon Adventure #1**

   Use the AISpace search tool to create a graph representation of the following maze navigation problem:



   In this maze it is possible for the agent to move between any two adjacent rooms unless they are blocked by a wall. Assume that *A* is the start state and *D* is the end state. Once you have completed your graph, use BFS or DFS to find a solution.

7. **Dungeon Adventure #2** Modify your graph from the previous exercise to represent the fact that there is a key at position *H* that must be collected before the agent can exit the maze:



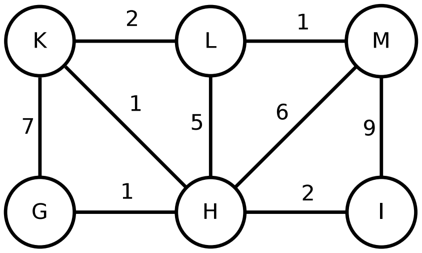
   Assume that picking up the key requires an action and the key cannot be dropped once it is picked up.

   Use DFS or BFS to find a solution.

   How many states would this problem have if the key can be dropped in any room after it has been picked up?

8. **Lowest-Cost-First Search**

Fill in the table below by tracing the execution of a minimum-cost search of the following graph:



Assume that G is the start state and L is the goal state. Use the following notation for path-nodes:

(State, Parent State, Path Cost)

Where “Parent State” refers to the state stored in the parent path node.

I’ve finished the first two rows to demonstrate the format.

Iteration #	Frontier	Explored
1.	(G, –, 0)	–
2.	(K, G, 7), (H, G, 1)	G
3.		
4.		
5.		
6.		
7.		
8.		
9.		
10.		

Examine the path nodes that you have created to confirm that they can be used to reconstruct the lowest-cost path to the goal.