CS444 Search Exercise

1. Consider formulating the following problem in terms of state search[1]:

   > A farmer with his wolf, goat and cabbage comes to the edge of a river he wishes to cross. There is a boat at the river's edge, but, of course, only the farmer can row. The boat also can carry only two things (including the rower) at a time. If the wolf is ever left alone with the goat, the wolf will eat the goat; similarly, if the goat is left alone with the cabbage, the goat will eat the cabbage. Devise a sequence of crossings of the river so that all four characters arrive on the other side with as few crossings as possible.

   (a) How many states would this problem have if we *didn't* need to worry about characters eating each other?

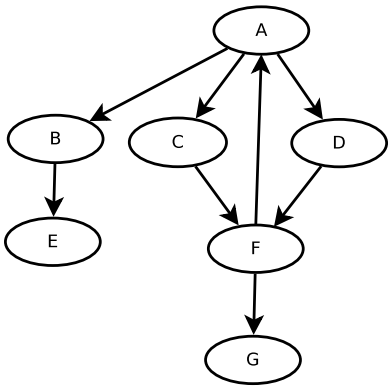
   (b) How many states does the problem have, given that we can't allow characters to eat each other?

   (c) On a separate sheet of paper, draw a graph representation of this problem. Indicate the start and goal states in your graph.

   (d) Find a solution to the problem by inspecting your graph. Mark the solution in your diagram.

[1] This question is adapted from *Artificial Intelligence*, Sixth Edition, 2009, George Luger.

The remaining 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.

2. Fill in the following table by tracing the execution of a depth-first search of the graph above. (Assume that when a node is expanded, its children are added to the frontier in alphabetical order.)

Iteration #	Frontier	Explored
1.		
2.		
3.		
4.		
5.		
6.		
7.		
8.		
9.		
10.		

3. Fill in the following table by tracing the execution of a breadth-first search of the graph above. (Assume that when a node is expanded, its children are added to the frontier in alphabetical order.)

Iteration #	Frontier	Explored
1.		
2.		
3.		
4.		
5.		
6.		
7.		
8.		
9.		
10.		

4. 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?