CS410: Artificial Intelligence 2021 Fall
Homework 1: Search Algorithms
Due date: 23:59:59 (GMT +08:00), October 10 2021

1. Consider the 8-queens problem. Your goal is to place 8 queens in a chessboard so that no two queens are in the same row, column, or diagonal. Recall that to formulate it as a search problem, we need to specify several components.

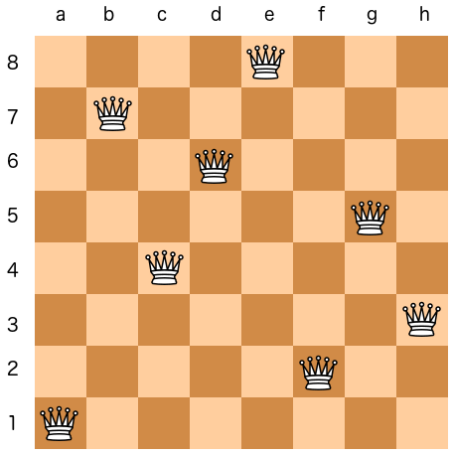


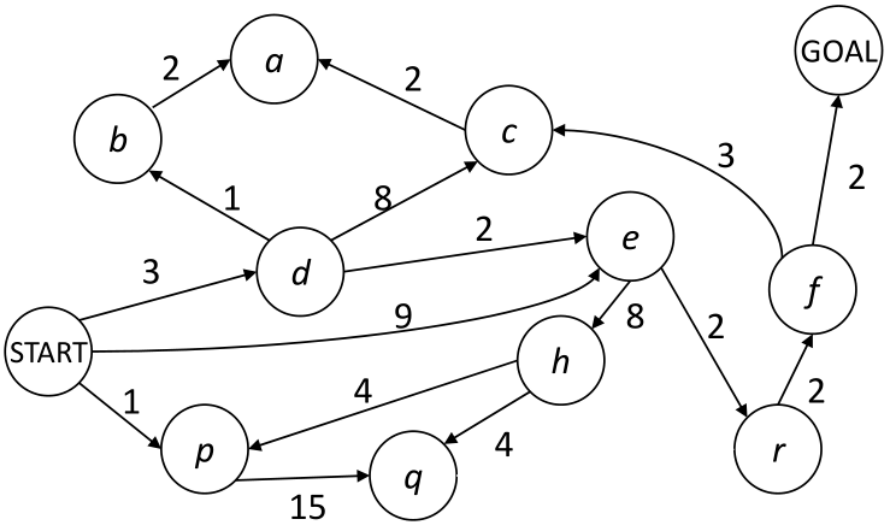
Figure 1: A feasible solution for the 8-queens problem.

   (a) Support we formulate the state space as the set of all arrangements of $0, 1, 2, \ldots, 8$ queens on the board. What would each component of the search problem be? How many states are there?

   (b) Give an alternative formulation which has a much smaller state space. List all the components of the search problem, and specify the number of states.

2. **Uniform cost graph search**. Consider the below state space graph. Perform UCS yourself and provide answers to the questions below regarding the nodes expanded during the search as well as the final path found by the algorithm. Remember that the search procedure should begin at node "START", and the goal state is node "GOAL". To break ties when adding nodes of equal cost to the fringe, follow the alphabetical order.

   Recall that UCS keeps track of the lowest cost, $c(v)$, to get from the start node to the node v.



   (a) What is the order of nodes expanded?

   (b) How many nodes are expanded?

   (c) What is the final path returned?

   (d) What is the length of path?

   (e) What is the cost of path?

3. **The graph-coloring problem.** Consider a graph with n nodes that has no multiple edges and loops. We want to color the nodes in this graph. We could only use 2 colors and we need to ensure that there is no edge linking any two nodes of the same color.

   (a) Please judge whether the following graphs could be colored as mentioned above. Write "Y" if you think one graph could be colored as mentioned above and write "N" otherwise.

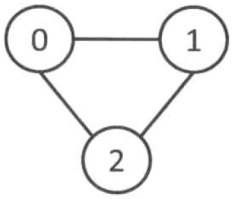


Figure 2: Graph 1.

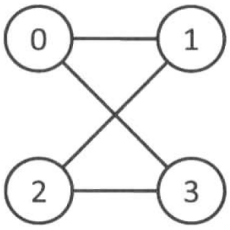


Figure 3: Graph 2.

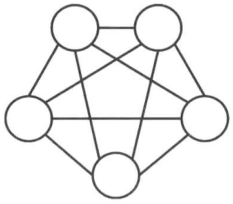


Figure 4: Graph 3.

   (b) Please briefly describe your findings about the properties of the graphs that could be colored as mentioned above.

   (c) We further assume the graph is connected. Let 1 and -1 be the two color values denoting the two colors respectively. Please fill in the following pseudocode for judging whether a graph could be colored as stated in the problem. (Algorithm 2 returns "True" if a graph could be colored as stated in the problem and returns "False" otherwise)

Algorithm 1 DFS

```
1: Input: node index v, color value c, color list color
2: color[v] ← c
3: for neighbor in get_neighbor_node(v) do
4:    if color[neighbor] == ____ then
5:       return False
6:    end if
7:    if color[neighbor] == 0 and not ________________ then
8:       return False
9:    end if
10: end for
11: return True
```

Algorithm 2 Color the Connected Graph

```
1: Initialize color list color ← (0)_{i=1}^{N}
2: if DFS(1,-1,color) then
3:    return ____
4: end if
5: return ____
```

(d) Please modify Algorithm2 to make Algorithm2 adapt to the graph which might be not connected. (Algorithm 3 returns "True" if a graph could be colored as stated in the problem and returns "False" otherwise)

Algorithm 3 Color the Graph (might be not connected)

```
1: Initialize color list color ← (0)_{i=1}^{N}
2: for node in get_all_node() do
3:    if color[node] == 0 then
4:       if not DFS(node,-1,color) then
5:          return ____
6:       end if
7:    end if
8: end for
9: return ____
```

4. **Explore the uniform-cost search in more depth.**

   (a) What are the differences between the uniform-cost search and Dijkstra's algorithm? (You could click here for the reference of Dijkstra's algorithm.)

   (b) If there are some arcs with negative costs in the graph, does the uniform-cost search still work? Give a concrete example to support your argument.

   (c) Introduce one algorithm which you know and is able to find the least-cost path in the graph containing the arcs with negative costs.