Graph Algorithm – Practice Problems - Solutions

1. A Hamiltonian path in a graph G=(V,E) is a simple that includes every vertex in V. Design an algorithm to determine if a directed acyclic graph (DAG) G has a Hamiltonian path. Your algorithm should run in O(V+E). Provide a written description of your algorithm including why it works, pseudocode and an explanation of the running time.

Solution:

Compute a topological sort and check if there is an edge between each consecutive pair of vertices in the topological order. If each consecutive pair of vertices are connected, then every vertex in the DAG is connected, which indicates a Hamiltonian path exists. Running time for the topological sort is O(V+E), running time for the next step is O(V) so total running time is O(V+E).

PSEUDOCODE:

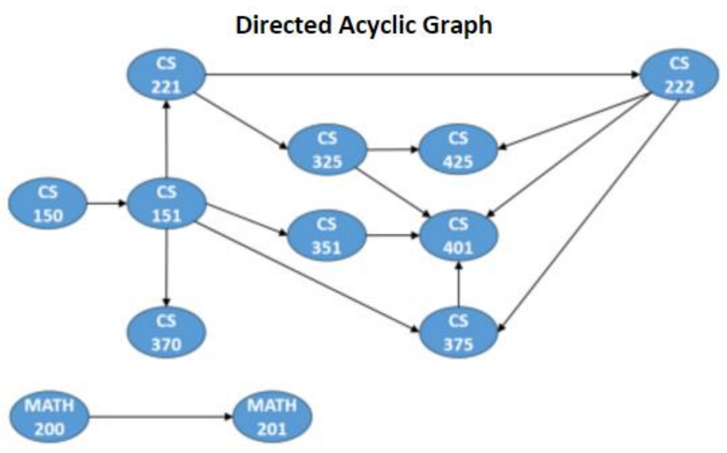
```
HAS_HAM_PATH(G)
    1. Call DFS(G) to computer finishing time v.f for each vertex v.
    2. As each vertex is finished, insert it into the front of the list
    3. Iterate each through the lists of vertices in the list
    4. If any pairs of consecutive vertices are not connected RETURN FALSE
    5. After all pairs of vertices are examined RETURN TRUE
```

2. Below is a list of courses and prerequisites for a factious CS degree.

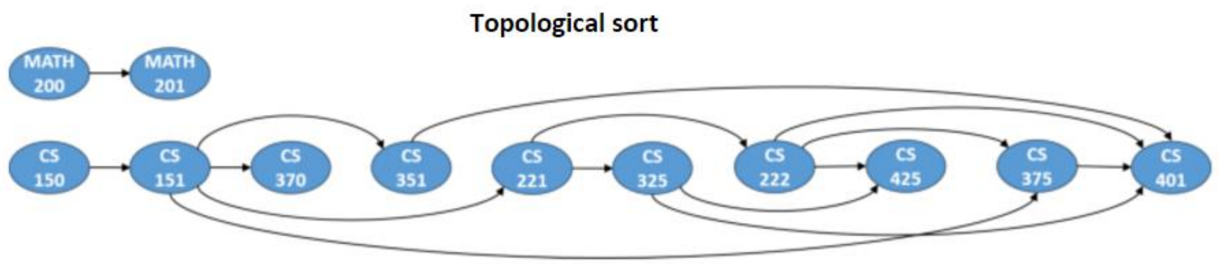
Course	Prerequisite
CS 150	None
CS 151	CS 150
CS 221	CS 151
CS 222	CS 221
CS 325	CS 221
CS 351	CS 151
CS 370	CS 151
CS 375	CS 151, CS 222
CS 401	CS 375, CS 351, CS 325, CS 222
CS 425	CS 325, CS 222
MATH 200	None
MATH 201	MATH 200

(a) Draw a directed acyclic graph (DAG) that represents the precedence among the courses.

Graph Algorithm – Practice Problems - Solutions



(b) Give a topological sort of the graph.



CS150 CS151 CS370 CS 351 CS221 CS325 CS222 CS425 CS375 CS 401 MATH 200 MATH 201 or

MATH 200 MATH 201 CS150 CS151 CS370 CS 351 CS221 CS325 CS222 CS425 CS375 CS 401

(c) Design an algorithm to find the longest path in an unweighted DAG where the length of a path is determined by the number of edges. What is the running time of the algorithm?

Algorithm to find the longest path in a DAG O(E+V)

1) Topologically sort the graph
2) Initialize the distances associated with vertices in the graph to 0. That is d(v)=0 for all v in G.
3) Start with the first vertex v in the topological sort.
    - For each u in Adj[v] set d(u) = max { d(u), d(v)+1 }
4) Repeat step 4 with the next vertex in the topological sorted order until all vertices have been examined.

(d) Find the longest path in the DAG from part b). What does this represent?

The length of the longest path is 5 which corresponds to the courses

CS150, CS151, CS 221, CS222, CS375, CS401.

Graph Algorithm – Practice Problems - Solutions

The length of the path+1 = 5+1 = 6 represented the minimum number of terms required to complete all courses with the given prerequisites.   (Note: this is often called the Critical Path)

(e) If you are allowed to take multiple courses at one time as long as there is no prerequisite conflict, find an order in which all the classes can be taken in the fewest number of terms.

Again several correct variations.

1st Term: CS 150, MATH 200

2nd Term: CS 151, CS 201

3rd Term: CS 221, CS 351, CS 370

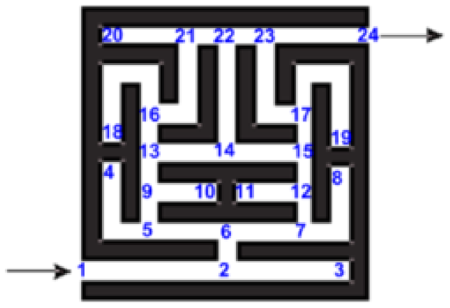
4th Term: CS 325, CS 222

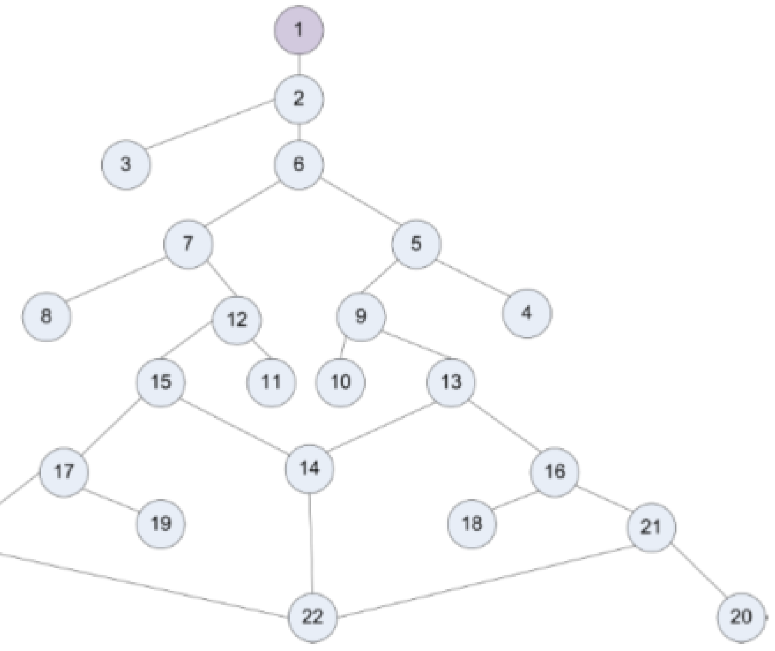
5th Term: CS 425, CS 375

6th Term: CS 401

3. One can model a maze by having a vertex for a starting point, a finishing point, dead ends and all the points in the maze where more than one path can be taken, and then connecting the vertices according to the paths in the maze.

a) Construct such a graph for the maze below:





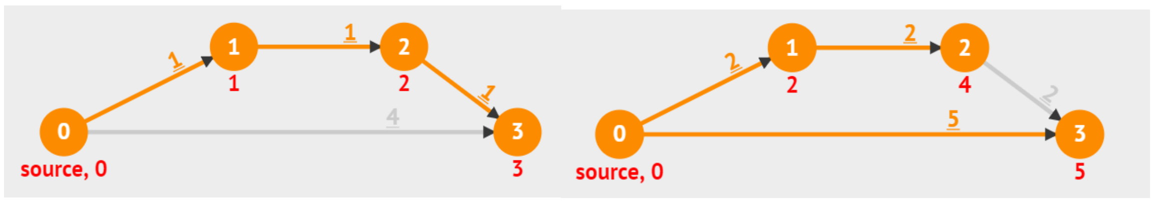
b) Design an algorithm to help you quickly determine an escape route for a maze such as the one above given that you have a graph representation. What is the running time of the algorithm?

You can use BFS running time O(V+E). This will give you the shortest path in terms of number of edges in the graph. Dijkstra's shortest path algorithm could also be used with edge weights that represent the length between intersections in the maze. If you do not have the lengths of the edges you could make them all 1 and then used Dijkstra's algorithm but this would be inefficient compared to BFS. DFS could also be used with a running time of O(V+E) but you may not find the shortest path out.

If calculated before entering the maze you could use the predecessor array to determine the quickest route out or the maze. The time is then O(V) or O(n) if |V|=n.

4. Consider an undirected graph G=(V,E) with nonnegative edge weights w(u,v)≥0. Suppose that you have computed a minimum spanning tree G, and that you have also computed shortest paths to all vertices from vertex s∈V. Now suppose each edge weight is increased by 1: the new weights w'(u,v) = w(u,v) + 1. Do the shortest paths change? Give an example where they change or prove they cannot change.

YES, the shortest path can change.



0
1
2
3
1
1
1
4
source, 0
1
2
3
0
1
2
3
2
2
2
5
source, 0
2
4
5