Jason Hsu

428005390

PA5 Report

1. Description of your implementation, C++ features used and assumptions on input data.

   The project uses several data structures, including queue, priority queue, hash table, and vector. The queue is used in the topological sort function to carry out topological sort. The vector is used in the vertex struct to store the adjacent list for the vertex. The hash table is used to store the information of the entire graph. (the element as the key, and its vertex as the data) The priority queue is used in the print topological sort function.

   As for the input data, we assume that the first element would be the vertex, and the rest elements in the line would be its adjacent list. Each line represent a single vertex.

2. Why does the topological sort algorithm use a queue? Can we use a stack instead?

   The topological sort function remove the element with indegree 0 and append it to a container. Queue is the best fit data structure for the algorithm for the ordering. Yes, we can also use a stack to carry out the topological sort. In fact, there are two ways to carry out the topological sort. One is suitable for a queue, the other is better using a stack. The second way to do the topological sort is as follow. First, we call the DFS on the graph, and then we output the vertices in decreasing order of their finishing time. (remove the newest element in the stack)

3. Can you explain why the algorithm detects cycles?

   The topological sort algorithm detects a cycle because only a directed acyclic graph is allowed to perform a topological sorting. If a graph contains

a cycle in it, it would cause an error calling topological sorting. Therefore, the algorithm detects cycles to prevent from such error.

4. What is the runtime for each function? Use the big-O notation and justify your answer.

    - buildGraph(): The function has a runtime of O(n), because we have to check through the entire hash table to see if the element already exist in the graph.
    - displayGraph() : The function also has a runtime of O(n), because it fo through the entire graph(hash table) and print out every single element in it.
    - At(): The function has a runtime of O(1). It simply returns the element at a particular address.
    - Size(): The function also has a runtime of O(1). It simple returns the size of the graph(hash).
    - Topological_sort(): The function has a runtime of O(v+e). v represents the number of vertices in the graph, while e represents the number of edges in the graph.
    - Computer_indegree(): The function has a runtime of O(n^2). We use a double for loop in the function to compute the indegree of a vertex.
    - Print_top_sort(): The function has a runtime of O(n). In the function, we go through the entire graph and append all the elements to a priority queue.

5. Case 1:

```
jasonhsu3322@LAPTOP-UG6DA0S0:/mnt/d/TAMU/Spring2021/CSCE221/PA5-TopSort$ ./a.out input.data
7 : 6
6 : 5
4 : 6 7
2 : 3 4 7
5 :
3 : 4
1 : 2 4 5
1 2 3 4 7 6 5
```

6. Case 2: A possible order is shown below at the end of the pic.

```
jasonhsu3322@LAPTOP-UG6DA0S0:/mnt/d/TAMU/Spring2021/CSCE221/PA5-TopSort$ ./a.out inputCourses.data
CSCE411 :
CSCE314 : CSCE315
CSCE315 :
CSCE313 :
CSCE312 : CSCE315
CSCE222 : CSCE221 CSCE411
CSCE221 : CSCE312 CSCE314 CSCE313 CSCE411
CSCE121 : CSCE221
CSCE222 CSCE121 CSCE221 CSCE312 CSCE314 CSCE313 CSCE411 CSCE315
```

7. Case 3:

```
jasonhsu3322@LAPTOP-UG6DA0S0:/mnt/d/TAMU/Spring2021/CSCE221/PA5-TopSort$ ./a.out inputLab.txt
LA127 :
LA126 :
LA32 : LA126 LA169
LA169 :
LA31 : LA32
LA22 : LA126 LA141
LA141 :
LA16 : LA32 LA127 LA141
LA15 : LA16 LA31
LA22 LA15 LA16 LA31 LA127 LA141 LA32 LA126 LA169
```

8. Case 4: A cycle is detect in the directed graph. The program would throw an error.

```
jasonhsu3322@LAPTOP-UG6DA0S0:/mnt/d/TAMU/Spring2021/CSCE221/PA5-TopSort$ ./a.out input-cycle.data
7 : 6
6 : 5
4 : 6 7
2 : 3 4 7
5 : 4
3 : 4
1 : 2 4 5
Cycle Detected.
```