# Mini Project: Graph Theory

#### Joshua Canlas

Electrical and Computer Engineering Stevens Institute of Technology Hoboken, NJ jcanlas1@stevens.edu

### I. DESIGN AND ALGORITHM

#### A. Pseudocode

# Algorithm 1 singleSourceWeightedShortestPath(source: char)

Create a dictionary of Node objects, where each Node is a vertex in the graph

Create a priority queue and make the first element the source vertex with distance zero

while queue is not empty do

Pop most recent vertex and distance from the priority queue

if vertex has been visited then

continue

end if

Mark the vertex as visited

for each neighbor of the current vertex do

if neighbor has been visited then

continue

end if

Calculate the new distance as current distance plus distance from vertex being evalu ated

**if** new distance < neighbor's distance from source **then** 

Update neighbor's distance from source Update neighbor's parent

Add neighbor and new distance to queue

end if

end for

end while

Print out vertices, distance from source, and corresponding parents

# **Algorithm 2** singleSourceUnweightedShortestPath(source: char)

The algorithm is the same as singleSourceWeightedShotest-Path(). The only difference is the calculation of the updated distance. The new distance is calculated as the number of edges from the source.

# Algorithm 3 minSpanningTree(source: char)

Create a dictionary of Node objects, where each Node is a vertex in the graph

Create a dictionary of vertices and distances, with the first element being the source and distance zero

Create an empty list to keep track of the minimum spanning tree

Initialize the minimum spanning tree total weight to be zero while dictionary of vertices and distances is not empty do

Find the minimum distance and corresponding vertex from the dictionary

if vertex has been visited then

continue

end if

Mark the vertex as visited

Append the vertex to the minimum spanning tree list

Update the total weight

for each neighbor to the current vertex do

if neighbor has been visited then

continue

end if

**if** neighbor is in the dictionary and its current distance > the distance listed in the dictionary **then** 

continue

end if

Update neighbor's distance

Update neighbor's parent

Add/update neighbor's current distance in the dictio-

nary

end for

end while

Print the minimum spanning tree Print the total weight of the tree

#### **Algorithm 4** maxSpanningTree(source: char)

The algorithm is the same as minSpanningTree(). The only difference is that we are looking for the maximum distance between vertices instead of minimum distance.

#### **Algorithm 5** DFS(visited: Set, vertex: char)

if vertex has not been visited then

Print the vertex

Add the vertex to the visited list

for each neighbor in the current vertex do

DFS(visited, neighbor)

end for

end if

# B. API Specifications

- Node()
- Graph(graph: Dict)
- Graph.singleSourceWeightedShortestPath(source: char)
- Graph.singleSourceUnweightedShortestPath(source: char)
- Graph.minSpanningTree(source: char)
- Graph.maxSpanningTree(source: char)
- Graph.BFS()

#### II. RESULTS

# A. Example Screenshots

Testing singleSourceWeightedShortestPath			
Vertex	Weighted Distance from A	Parent	
Α	0	Α	
В	5	Α	
C	3	Α	
D	10	С	
E	6	В	
F	8	В	
G	9	F	

Fig. 1. singleSourceWeightedShortestPath test

Testing singleSourceUnweightedShortestPath			
Vertex	Unweighted Distance from B	Parent	
Α	3	D	
В	0	В	
C	1	В	
D	2	С	
E	1	В	
F	1	В	
G	2	F	

Fig. 2. singleSourceUnweightedShortestPath test

```
Testing minSpanningTree...
Minimum spanning tree: ['A', 'B', 'F', 'G', 'C', 'E', 'I', 'H', 'D', 'J']
Minimum spanning tree total weight: 25
```

Fig. 3. minSpanningTree test

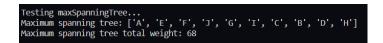


Fig. 4. maxSpanningTree test

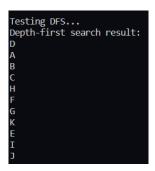


Fig. 5. DFS test

#### B. Additional Information

The test cases for this mini project have been hard-coded. Therefore, if the user wants to change the graph, they will need to update the code.

I have updated my branch on GitHub (Canlas9875). The Python script is located under the GraphTheory folder (Canlas9875\_GraphAlgorithms.py). Please contact me if you have any questions or feedback.