**Project 3 Report**

**Group #3**

Language: Java

Da-El Kim

Hang Phan

Alibek Zeinolla

Olufisayo Ayodele

Alikhan Adilbekov

**Outline**

Interpretation of the Given Problem…………………………………….3

Methodology of the Solution……………………………………………4

Experimental Results……………………………………………………6

Conclusion………………………………………………………………8

**Interpretation of the Given Problem**

There are many different algorithms that can be used to find the shortest path between two nodes. Dijkstra's algorithm is an algorithm for finding the shortest paths between nodes in a graph, which may represent, for example, road networks. The Bellman–Ford algorithm is an algorithm that computes shortest paths from a single source vertex to all of the other vertices in a weighted digraph.

In this project, we were asked to implement and test a greedy algorithm (Dijkstra’s Algorithm) and a dynamic programming algorithm (Bellman Ford). Graphs are broadly used in modeling real world applications. Our real-world application in this case would be finding the shortest paths between buildings in the University of Central Arkansas and the Math and Science building.

Our objectives in this project were to simplify and solve real world problems (finding shortest path from one building to another), apply an existing algorithm into real world application (Dijkstra and Bellman Ford Algorithms), design reasonable software solutions (output would be shown below), implement and verify potential solutions, and collaborate with team members (because team work is a vital part of a programmer’s job).

**Methodology of the Solution**

As learnt from our previous two projects, using object-oriented programming is the most efficient way to achieve the best solution compelled with reusability, and that’s why we chose Java. First, a Node class was created to store the building names and length to a key (another building). The class has methods that are described in more detail below:

*Node class*:

**public** Node(): Constructor. It initializes all the private variables.

**public** **void** addNode(String name, **int** length): Adds the name of a building and a length of the path of the building from a key

Input: name of the building, length of the building from the key.

Process: put the input in a map(node) and two separate ArrayLists.

Output: no output.

**public** ArrayList<String> getVertex(): Getter for the vertex (building name)

**public** ArrayList<String> getEdgeLength(): Getter for the edgeLength (building length from key)

Next, we created a ShortestPath class that ultimately implements Dijkstra’s and Bellman Ford’s algorithm. The details and processes about the class and its methods are further explained below:

*ShortestPath class*:

**public** ShortestPath(): Constructor. It initializes all the private variables prev and dist. prev is initialized to null and dist is initialized to ∞ (maximum integer value)

**public** **void** getAllKey(): Getter for all keys (building names).

Input: no input.

Process: a for loop to store all the keys of the TreeMap in an ArrayList

Output: no output.

**public** String toString():

**public** **void** initialize(String myFileName): This method reads from the text file

Input: file name.

Process: read from the file and send each individual line to the splitLine method to be organized.

Output: no output.

**public** **void** splitLine(String line): This method organizes each line sent from the text file. It extracts only the data that we need from each line in the text file.

Input: a line from the text file.

Process: split the given line and extract important data like building names and distance.

Output: no output.

**public** **void** dijkstra(String building): This method implements Dijkstra's single source shortest path algorithm for a graph represented using adjacency matrix

Input: building name.

Process: set the distance from key building to 0, go through the dijkstra algorithm process, for each node u, dist[u] is set to its distance from s.

Output: no output.

**public** String deletemin(String[] queueH):

Input: priority queue.

Process: get all index of queueH and compare the dist from it index in allKey, to find min and return the key of the min.

Output: building to be deleted, null if otherwise.

**public** **void** shortestPaths(String buidling): Bellman Ford’s algorithm

Input: building name.

Process: The first for loop sets the distance to each vertex in the graph to infinity. This is later changed for the source vertex to equal zero. Also, in that first for loop, the prev value for each vertex is set to nothing. This value is a pointer to a predecessor vertex so that we can create a path later. The next for loop simply goes through each edge (u, v) in E and updates it. This process is done |V| - 1 times

Output: no output

**public** **void** update(String u, String v, **int** length): Calculation of the temporary distances of all neighbor nodes of the active node by summing up its distance with the weights of the edges. If such a calculated distance of a node is smaller as the current one, update the distance and set the current node as antecessor.

Then we wrote a Tester class to test the program that we’ve written, and the results of the testing are shown in the Experimental Results page.

**Experimental Results**

*First reads the file*

!!!!!!!!!!DONE READING FILE!!!!!!!!!!

*Display extracted data with overridden toString method*

Bear village Apt.: Brewer-Hegeman=350,

Brewer-Hegeman: Bear village Apt.=350,Oak Tree Apt=40,Student Health=200,New Business=20,

Burdick: Math-Comp Science=30,Speech Language=100,McAlister Hall=200,Maintenance=300,Torreyson Library=80,

College Square: Lewis Science=200,Prince Center=300,

Fine Art: McAlister Hall=180,Police Dept=50,Old Main=90,Student=80,

Lewis Science: Math-Comp Science=150,Speech Language=250,College Square=200,

Maintenance: Oak Tree Apt=160,Speech Language=120,McAlister Hall=150,Burdick=300,New Business=150,Wingo=100,

Math-Comp Science: Lewis Science=150,Prince Center=80,Burdick=30,Torreyson Library=40,

McAlister Hall: Fine Art=180,Burdick=200,Old Main=100,Maintenance=150,Student=100,Wingo=50,

New Business: Oak Tree Apt=30,Brewer-Hegeman=20,Maintenance=150,Student=110,Wingo=50,

Oak Tree Apt: Brewer-Hegeman=40,Maintenance=160,New Business=30,

Old Main: McAlister Hall=100,Police Dept=200,Fine Art=90,Torreyson Library=30,

Police Dept: Fine Art=50,Student Health=100,Prince Center=100,Old Main=200,

Prince Center: Math-Comp Science=80,Police Dept=100,College Square=300,Torreyson Library=30,

Speech Language: Lewis Science=250,Burdick=100,Maintenance=120,

Student: McAlister Hall=100,Fine Art=80,Student Health=50,New Business=110,Wingo=100,

Student Health: Police Dept=100,Brewer-Hegeman=200,Student=50,

Torreyson Library: Math-Comp Science=40,Prince Center=30,Burdick=80,Old Main=30,

Wingo: McAlister Hall=50,Maintenance=100,New Business=50,Student=100,

*Then we test the Dijkstra method to get the shortest path of all buildings in the map from the Computer Science Building*

Dijkstra:

distance from Math-Comp Science to Bear village Apt. is 640 previous building is Brewer-Hegeman

distance from Math-Comp Science to Brewer-Hegeman is 290 previous building is New Business

distance from Math-Comp Science to Burdick is 30 previous building is Math-Comp Science

distance from Math-Comp Science to College Square is 350 previous building is Lewis Science

distance from Math-Comp Science to Fine Art is 160 previous building is Old Main

distance from Math-Comp Science to Lewis Science is 150 previous building is Math-Comp Science

distance from Math-Comp Science to Maintenance is 250 previous building is Speech Language

distance from Math-Comp Science to Math-Comp Science is 0 previous building is null

distance from Math-Comp Science to McAlister Hall is 170 previous building is Old Main

distance from Math-Comp Science to New Business is 270 previous building is Wingo

distance from Math-Comp Science to Oak Tree Apt is 300 previous building is New Business

distance from Math-Comp Science to Old Main is 70 previous building is Torreyson Library

distance from Math-Comp Science to Police Dept is 170 previous building is Prince Center

distance from Math-Comp Science to Prince Center is 70 previous building is Torreyson Library

distance from Math-Comp Science to Speech Language is 130 previous building is Burdick

distance from Math-Comp Science to Student is 240 previous building is Fine Art

distance from Math-Comp Science to Student Health is 270 previous building is Police Dept

distance from Math-Comp Science to Torreyson Library is 40 previous building is Math-Comp Science

distance from Math-Comp Science to Wingo is 220 previous building is McAlister Hall

*We do the same for the Bellman Ford algorithm and as expected, we get the same results*

Bellman Ford:

distance from Math-Comp Science to Bear village Apt. is 640. previous building is Brewer-Hegeman

distance from Math-Comp Science to Brewer-Hegeman is 290. previous building is New Business

distance from Math-Comp Science to Burdick is 30. previous building is Math-Comp Science

distance from Math-Comp Science to College Square is 350. previous building is Lewis Science

distance from Math-Comp Science to Fine Art is 160. previous building is Old Main

distance from Math-Comp Science to Lewis Science is 150. previous building is Math-Comp Science

distance from Math-Comp Science to Maintenance is 250. previous building is Speech Language

distance from Math-Comp Science to Math-Comp Science is 0. previous building is null

distance from Math-Comp Science to McAlister Hall is 170. previous building is Old Main

distance from Math-Comp Science to New Business is 270. previous building is Wingo

distance from Math-Comp Science to Oak Tree Apt is 300. previous building is New Business

distance from Math-Comp Science to Old Main is 70. previous building is Torreyson Library

distance from Math-Comp Science to Police Dept is 170. previous building is Prince Center

distance from Math-Comp Science to Prince Center is 70. previous building is Torreyson Library

distance from Math-Comp Science to Speech Language is 130. previous building is Burdick

distance from Math-Comp Science to Student is 240. previous building is Fine Art

distance from Math-Comp Science to Student Health is 270. previous building is Police Dept

distance from Math-Comp Science to Torreyson Library is 40. previous building is Math-Comp Science

distance from Math-Comp Science to Wingo is 220. previous building is McAlister Hall

*Then we switch it up a little bit to get the shortest path of all buildings in the map from the Lewis Science Center using Bellman Ford algorithm*

Bellman Ford:

distance from Lewis Science to Bear village Apt. is 790. previous building is Brewer-Hegeman

distance from Lewis Science to Brewer-Hegeman is 440. previous building is New Business

distance from Lewis Science to Burdick is 180. previous building is Math-Comp Science

distance from Lewis Science to College Square is 200. previous building is Lewis Science

distance from Lewis Science to Fine Art is 310. previous building is Old Main

distance from Lewis Science to Lewis Science is 0. previous building is null

distance from Lewis Science to Maintenance is 370. previous building is Speech Language

distance from Lewis Science to Math-Comp Science is 150. previous building is Lewis Science

distance from Lewis Science to McAlister Hall is 320. previous building is Old Main

distance from Lewis Science to New Business is 420. previous building is Wingo

distance from Lewis Science to Oak Tree Apt is 450. previous building is New Business

distance from Lewis Science to Old Main is 220. previous building is Torreyson Library

distance from Lewis Science to Police Dept is 320. previous building is Prince Center

distance from Lewis Science to Prince Center is 220. previous building is Torreyson Library

distance from Lewis Science to Speech Language is 250. previous building is Lewis Science

distance from Lewis Science to Student is 390. previous building is Fine Art

distance from Lewis Science to Student Health is 420. previous building is Police Dept

distance from Lewis Science to Torreyson Library is 190. previous building is Math-Comp Science

distance from Lewis Science to Wingo is 370. previous building is McAlister Hall

**Conclusion**

In conclusion, we find out that Dijkstra and Bellman Ford algorithms are very useful tools for finding the shortest path between buildings. Their real-world applications are enormous. We found out that using shortest path algorithm, suppose that any geographical map is a GRAPH. Now locations in the map are our VERTICES in algorithm. And road between locations are EDGES. Now WEIGHTS OF EDGES here are distance between those two locations. So, by this tech companies can make a GPS system which will guide you to the locations.

Researching Dijkstra's algorithm, we found out that it is widely used in the routing protocols required by the routers to update their forwarding table. The algorithm provide the shortest cost path from source router to other routers in the network. Bellman-Ford is capable of handling graphs that contain negative edge weights, so it is more versatile.

In this project, we simplified and solved a real-world problem by applying an existing algorithm into a real-world application. We designed reasonable software solutions, implemented and verified potential solutions to them. Team work was a very important aspect of our project because we found that work like this is better split among members. Individually, we couldn’t exactly figure it out but brainstorming as a group gave us an edge to solving the problem.

In the future, now that we have learnt different algorithms in this class, we are versatile and can choose any one that suits whatever project we are working on. Like using Bellman Ford algorithm when there is sure to be negative nodes or using Dijkstra algorithm when there is a time constraint because it is faster.

.