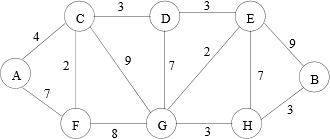
CS 325 – Analysis of algorithm

**Problem 1.** *(10 points)* A region contains a number of towns connected by roads. In the graph below, each town is labeled with a circle and each road is labeled with the average number of hours required for a truck to travel between the cities. Suppose that you work for a large retailer that has decided to place a distribution center in town G. (While this problem is small, you want to devise a method to solve much larger problems).



**(a) Which algorithm would you recommend be used to find the fastest route from the distribution center to each of the towns? Demonstrate how it would work on the example above if the distribution center is placed at town G. Show the resulting routes.**

Solution:

Because it is a single source shortest path problem with undirected path, it is ok to use Dijkstra algorithm.

Demonstrate: (It starts from G)

Step 1:

|  |  |  |  |
| --- | --- | --- | --- |
|  | Start | Distance | Previous |
| A |  |  |  |
| B |  |  |  |
| C |  | 9 | G |
| D |  | 7 | G |
| E |  | 2 | G |
| F |  | 8 | G |
| G | T | 0 |  |
| H |  | 3 | G |

Step 2: (Start from E)

|  |  |  |  |
| --- | --- | --- | --- |
|  | Start | Distance | Previous |
| A |  |  |  |
| B |  | 11 | E |
| C |  | 9 | G |
| D |  | 5 | E |
| E | T | 2 | G |
| F |  | 8 | G |
| G | T | 0 |  |
| H |  | 3 | G |

Step3: (Start from H)

|  |  |  |  |
| --- | --- | --- | --- |
|  | Start | Distance | Previous |
| A |  |  |  |
| B |  | 6 | H |
| C |  | 9 | G |
| D |  | 5 | E |
| E | T | 2 | G |
| F |  | 8 | G |
| G | T | 0 |  |
| H | T | 3 | G |

Step 4: Start from D (Do nothing)

|  |  |  |  |
| --- | --- | --- | --- |
|  | Start | Distance | Previous |
| A |  |  |  |
| B |  | 6 | H |
| C |  | 8 | D |
| D | T | 5 | E |
| E | T | 2 | G |
| F |  | 8 | G |
| G | T | 0 |  |
| H | T | 3 | G |

Step 5: Start from F

|  |  |  |  |
| --- | --- | --- | --- |
|  | Start | Distance | Previous |
| A |  | 15 | F |
| B |  | 6 | H |
| C |  | 8 | D |
| D | T | 5 | E |
| E | T | 2 | G |
| F | T | 8 | G |
| G | T | 0 |  |
| H | T | 3 | G |

Step 6: Start from C

|  |  |  |  |
| --- | --- | --- | --- |
|  | Start | Distance | Previous |
| A |  | 12 | C |
| B |  | 6 | H |
| C | T | 8 | D |
| D | T | 5 | E |
| E | T | 2 | G |
| F | T | 8 | G |
| G | T | 0 |  |
| H | T | 3 | G |

Step 7: Start from B (Do nothing)

|  |  |  |  |
| --- | --- | --- | --- |
|  | Start | Distance | Previous |
| A |  | 12 | C |
| B | T | 6 | H |
| C | T | 5 | D |
| D | T | 5 | E |
| E | T | 2 | G |
| F | T | 8 | G |
| G | T | 0 |  |
| H | T | 3 | G |

Step 8: Start from A (Do nothing)

|  |  |  |  |
| --- | --- | --- | --- |
|  | Start | Distance | Previous |
| A | T | 12 | C |
| B | T | 6 | H |
| C | T | 8 | D |
| D | T | 5 | E |
| E | T | 2 | G |
| F | T | 8 | G |
| G | T | 0 |  |
| H | T | 3 | G |

Result: A has the largest distance from G

(b) Suppose one ”optimal” location (maybe instead of town G) must be selected for the distribution center such that it minimizes the time travelled to the farthest town. Devise an algorithm to solve this problem given an arbitrary road map. Analyze the time complexity of your algorithm when there are t possible towns (locations) for the distribution center and r possible roads.

Idea:

I will use Dijkstra algorithm to search the distance for each place. Then, I will do it for t times because I want to know the farthest distance of each distribution center position. After that, I would store them into an array then find out the minimal farthest distance.

Time complexity:

The time complexity of Dijkstra algorithm is O(). Then, I need to do it t time. The time complexity becomes O(). Finding out the farthest distance from a particular distribution center takes O(t). Finding out the minimal one takes O(t). So, the total time complexity is O(). We can say it is O().

(c) In the above graph which is the “optimal” town to locate the distribution center?

1. Start from A

|  |  |  |  |
| --- | --- | --- | --- |
|  | Start | Distance | Previous |
| A | T | 0 |  |
| B | T | 18 | H |
| C | T | 4 | A |
| D | T | 7 | C |
| E | T | 10 | D |
| F | T | 6 | C |
| G | T | 12 | E |
| H | T | 15 | G |

2. Start from B

|  |  |  |  |
| --- | --- | --- | --- |
|  | Start | Distance | Previous |
| A | T | 18 | C |
| B | T | 0 |  |
| C | T | 14 | D |
| D | T | 11 | E |
| E | T | 8 | G |
| F | T | 14 | G |
| G | T | 6 | H |
| H | T | 3 | B |

3. Start from C

|  |  |  |  |
| --- | --- | --- | --- |
|  | Start | Distance | Previous |
| A | T | 4 | C |
| B | T | 14 | H |
| C | T | 0 |  |
| D | T | 3 | C |
| E | T | 6 | D |
| F | T | 2 | C |
| G | T | 8 | E |
| H | T | 11 | G |

4. Start from D

|  |  |  |  |
| --- | --- | --- | --- |
|  | Start | Distance | Previous |
| A | T | 7 | C |
| B | T | 11 | H |
| C | T | 3 | D |
| D | T | 0 |  |
| E | T | 3 | D |
| F | T | 5 | C |
| G | T | 5 | E |
| H | T | 8 | G |

5. Start from E

|  |  |  |  |
| --- | --- | --- | --- |
|  | Start | Distance | Previous |
| A | T | 10 | C |
| B | T | 8 | H |
| C | T | 6 | D |
| D | T | 3 | E |
| E | T | 0 |  |
| F | T | 8 | C |
| G | T | 2 | E |
| H | T | 5 | G |

6. Start from F

|  |  |  |  |
| --- | --- | --- | --- |
|  | Start | Distance | Previous |
| A | T | 6 | C |
| B | T | 14 | H |
| C | T | 2 | F |
| D | T | 5 | C |
| E | T | 8 | D |
| F | T |  |  |
| G | T | 8 | F |
| H | T | 11 | G |

7. Start from G

|  |  |  |  |
| --- | --- | --- | --- |
|  | Start | Distance | Previous |
| A | T | 12 | C |
| B | T | 6 | H |
| C | T | 8 | D |
| D | T | 5 | E |
| E | T | 2 | G |
| F | T | 8 | G |
| G | T | 0 |  |
| H | T | 3 | G |

8. Start from H

|  |  |  |  |
| --- | --- | --- | --- |
|  | Start | Distance | Previous |
| A | T | 15 | C |
| B | T | 3 | H |
| C | T | 11 | D |
| D | T | 8 | E |
| E | T | 5 | G |
| F | T | 11 | G |
| G | T | 3 | H |
| H | T | 0 |  |

The farthest distance of {A,B,C,D,E,F,G,H} = {18,18,14,11,10,14,13,15}

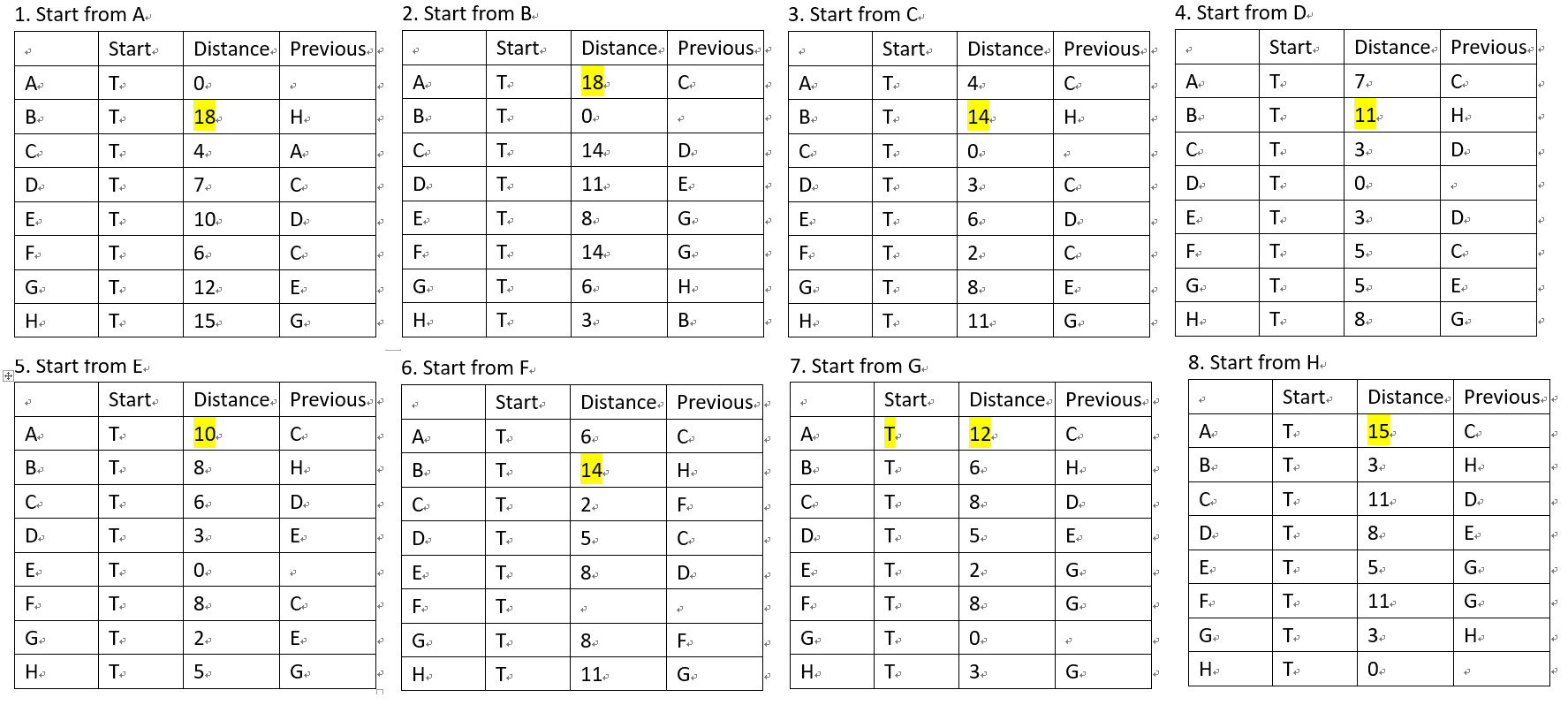
The minimal one is 10. The distribution center should be located at E.

**Now suppose you can build two distribution centers. Where would you place them to minimize the time travelled to the farthest town? Describe an algorithm to solve this problem given an arbitrary road map..**

Even though we have two distribution centers, we still need to know all the distances from each possible distribution center. So, the first step is just same as the previous question. Then, I will build a 8 X 28 matrix, a variable to store the current maximum distance, and an array to store the two distribution centers we pick. The 8 x 28 matrix is a two dimension matrix. 8 means that we have 8 places, A,B,C,D,E,F,G,H. 28 means that we could have 28 choices if two distribution centers are allowed. (8\*7/2=28) Then, we will do the calculation and store the maximum distance into a variable based on the table we do from the previous questions. Comparing the maximum distance with the current two distribution centers is the next step. If there is one distance larger than the variable we store in the last step, we can just ignore these two distribution centers and pick the next two to do calculation. As we update the variable, we also need to update the two distribution centers we pick into an array. As long as we finish the iteration, we can solve this problem.

**The time complexity** of this algorithm is also O(). Even though we need to compare the maximum distance after we do Dijkstra algorithm, but its running time is much slower than Dijkstra algorithm. So, we can say the time complexity is still O().

**(e) In the above graph what are the “optimal” towns to place the two distribution centers?**



|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **AB** | **AC** | **AD** | **AE** | **AF** | **AG** | **AH** | **BC** | **BD** | **BE** | **BF** | **BG** | **BH** |
| **A** | **0** | **0** | **7** | **0** | **0** | **0** | **0** | **4** | **7** | **10** | **6** | **12** | **15** |
| **B** | **0** | **14** | **11** | **8** | **14** | **6** | **3** | **0** |  |  | **0** |  |  |
| **C** | **4** |  |  | **4** |  | **4** | **4** | **0** |  |  | **2** |  |  |
| **D** | **7** |  |  | **3** |  | **5** | **7** | **3** |  |  | **5** |  |  |
| **E** | **8** |  |  | **0** |  | **2** |  | **6** |  |  | **8** |  |  |
| **F** | **6** |  |  | **6** |  | **6** |  | **2** |  |  |  |  |  |
| **G** | **6** |  |  | **2** |  | **0** |  | **6** |  |  |  |  |  |
| **H** | **3** |  |  | **5** |  | **3** |  | **3** |  |  |  |  |  |
| **M** | **8** |  |  | **8** |  | **6** |  | **6** |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **CD** | **CE** | **CF** | **CG** | **CH** | **DE** | **DF** | **DG** | **DH** | **EF** | **EG** | **EH** | **FG** | **FH** | **GH** |
| **A** | **0** | **4** | **4** | **4** | **4** | **7** | **6** | **7** | **7** | **6** | **10** | **10** | **6** | **6** | **12** |
| **B** | **11** | **8** | **14** | **6** | **3** |  |  |  |  |  |  |  |  |  |  |
| **C** |  |  |  | **0** | **0** |  |  |  |  |  |  |  |  |  |  |
| **D** |  |  |  | **3** | **3** |  |  |  |  |  |  |  |  |  |  |
| **E** |  |  |  | **2** | **5** |  |  |  |  |  |  |  |  |  |  |
| **F** |  |  |  | **2** | **2** |  |  |  |  |  |  |  |  |  |  |
| **G** |  |  |  | **0** | **3** |  |  |  |  |  |  |  |  |  |  |
| **H** |  |  |  | **3** | **0** |  |  |  |  |  |  |  |  |  |  |
| **M** |  |  |  | **6** | **5** |  |  |  |  |  |  |  |  |  |  |

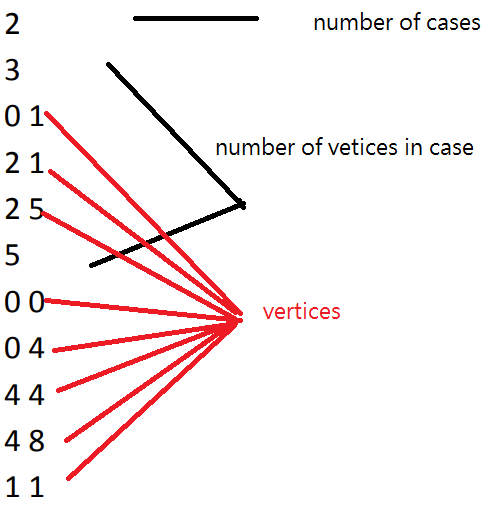
The optimal towns we pick would be **C and H**. The maximum distance is 5.

**Problem 2.** *(20 points)* **Euclidean MST Implementation**

A verbal describe of your algorithm and data structures

Date process and designed structure:

Storing data into two different arrays, number of vertices and vertices.

p

After we store vertices into an array called All\_vertice, we also pick up the first element as our starting points and store it into another array called Parents, which stores the vertices we have already chosen. Then, based on elements in the Parents array, we start to consider the next vertex we are going to choose. What we are doing here is pick up the vertex from All\_vertice array having a shortest distance with all the elements in the chosen array, Parents. Once we find out the vertex, we just pop this out and store it into Parents array. Step by Step, as long as there is no vertex in the array, All\_vertice, we finish our goal.

Theoretical running time

The time complexity of function, distance is O(1)

The running time of function, pick\_node is based on the length of parent and length of graph. And both depend on the size of vertices we have in this problem. Assume we have n vertices in this case, the running time of my pick\_node function would be O(k (n-k)). Since we have to compare all the vertices in our array, we can say the time complexity of my code is O()