Middle East Technical University Department of Computer Engineering

CENG 315

ALGORITHMS

Fall 2017

Take Home

Due date: 24th December 2017, Sunday, 23:55

1 Problem Definition

In this assignment, you will implement a solution to the following problem where input is a graph. Given a connected positive weighted undirected graph $G(V, E)$ with vertices $w_1, w_2 \in V$ and $d_1, ..., d_k \in V$ where k is always even, find the shortest total length of paths that connect $k/2$ d vertices to w_1 and remaining $k/2$ d vertices to w_2. The solution is the total length of the paths and the assignments between w_j's and d_i's. If there is more than one solution, one of the assignments is enough as a result.

Informally, you have two warehouses w_1 and w_2. Each warehouse have $k/2$ trucks that can only carry the load for one city and you have k destination cities $d_1, ...d_k$. The goal is to find the shortest possible total distance the trucks will travel and the assignment of destination cities to warehouses in this optimal solution.

Note that the total travel cost is computed by simply summing the lengths of the paths taken individually by each truck. Therefore, you shall ignore path overlaps across the pairs of warehouse and destination found by your algorithm

2 Specifications

2.1 Libraries and Complexity

You can use the *C++* standard library in your program and nothing else. The expected complexity of your solution is $O((E + V) \cdot log(V))$.

2.2 Input Specifications

Your program will get a file name as parameter. The file for the graph shown in figure 1 is as follows:

```
10
4
5 7
1 4 6 8
0 43 0 46 15 0 0 0 0 0
43 0 0 50 0 36 43 0 0 0
0 0 0 0 0 0 0 49 13 0
```

```
46 50 0 0 0 0 0 0 0 0
15 0 0 0 0 0 0 0 46 45
0 36 0 0 0 0 20 0 0 0
0 43 0 0 0 20 0 0 0 0
0 0 49 0 0 0 0 0 0 0
0 0 13 0 46 0 0 0 0 0
0 0 0 0 45 0 0 0 0 0
```

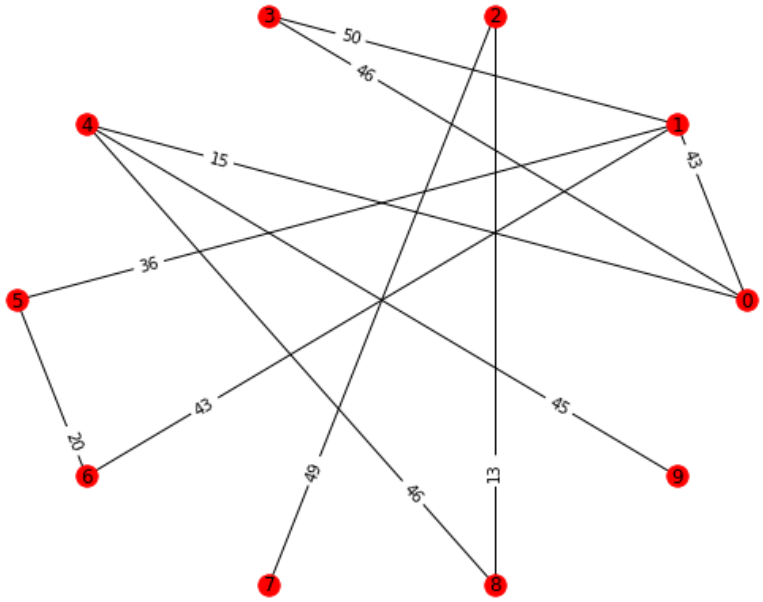

Figure 1: Sample input visualization

The first line of the file gives the total number of cities n, including warehouses and the destinations. The second line is the number of destinations. The third line is the indexes for two warehouses. The fourth line gives the indexes for the destination cities. Both indexes are zero based. Rest of the file represents an n by n matrix that represents the distances between cities, where 0 means no connection. You can assume there will be at most 10.000 cities.

2.3 Output Specifications

Your program should write the result to the standard output. First line must be the total length of the paths for a given input. The following lines must show the assignments of warehouses to each city in ascending order.

```
226
1 5
4 7
6 5
8 7
```

2.4 Execution

Your make file must generate the executable `runTH` . Your program will be run with the following command.

```
./runTH fileName
```

2.5 Submission

Your submission should be a tar file, only containing your code and the `Makefile` to compile your code in its root. You can do this with a command such as this: `tar -cvf TH.tar *.cpp *.hpp makefile`

3 Regulations

1. **Programming Language:** You must code your program in `C++`. Your submission will be compiled with `g++` on department lab machines.
2. **Late Submission:** There is no late submission.
3. **Evaluation:** Black box evaluation method is going to be used, therefore you need to be careful about input/output specifications. You should not print any unnecessary characters and/or white spaces. Missing make files will be evaluated as compile errors. Wrong executable names or wrong parameterization will be evaluated as if no output is produced.
4. **Cheating:** In case of cheating, the university regulations will be applied.