Due: 22nd of August 2016 at 11:59pm

COMP 2007 – Assignment 1

All submitted work must be done individually without consulting someone elses solutions in accordance with the Universitys Academic Dishonesty and Plagiarism policies.

IMPORTANT! Questions 1 and 2a-c should be submitted via Blackboard as pdf (no handwriting!). The implementation required for Question 2d should be done in Ed, and submitted via Ed.

Questions

1. Sort the following functions by order of growth from slowest to fastest (big-O notation). For each pair of adjacent functions in your list, please write one sentence describing (informally is fine) why you ordered them as you did.

$$7n^3 - 10n, 4n^2, n, n^{8621909}, 3n, 2^{\log \log n}, n \log n, 6n \log n, n!, 1.1^n$$

[20 points]

2. Consider the following variant of the MST problem. We are given an undirected graph $G = (V, E)$ with n vertices and m edges, a positive (not necessarily unique) edge cost c_e for each edge in E, and a subset of vertices $A \subset V$. Suppose that V represents a set of computers and A is the subset of these computers that have a single network port to connect to another computer. We would like to find the cheapest way to design a computer network that will connect all computers.

   The abstract problem we are interested in solving is to find a subset $X \subseteq E$ of edges of minimum cost such that (V, X) is connected and $\deg_X(u) = 1$ for all $u \in A$.

   Your task is to design an algorithm that solves this problem in $O(m \log n)$ time. Your solution must include:

   (a) Description of how your algorithm works ("in plain English"). [20 points]

   (b) Argue why your algorithm is correct. [15 points]

   (c) Prove an upper bound on the time complexity of your algorithm. [5 points]

(d) Implement your algorithm (in Ed) and test it on the the following instances `Graph-8, Graph-250, Graph-1000`. Each instance is given in a text file using the following format:

```
n
m
A
vertexId vertexId weight
vertexId vertexId weight
...
```

For example, the following text describes the instance depicted in Fig. 1. Note that the vertices are numbered from 0 to $n-1$.

```
4
5
0 2
0 1 0.6
0 2 0.2
0 3 0.5
1 3 0.8
2 3 0.3
```

Your program should read input from standard input, and calculate the weight of the spanning tree generated by your algorithm. Your program does not need to return the actual edges, just print out the total weight rounded to **2 decimal places** to standard output. A scaffold is provided for Python 3 for you, but if you would like to use any other language, simply delete a1.py and replace with your file, and edit build.sh and run.sh to compile (if required) and run your code respectively.

Your code will not be benchmarked or tested for time complexity, but for full points it must be able to run instances similar to "Graph-1000", and each test will time out after 5 seconds. [40 points]

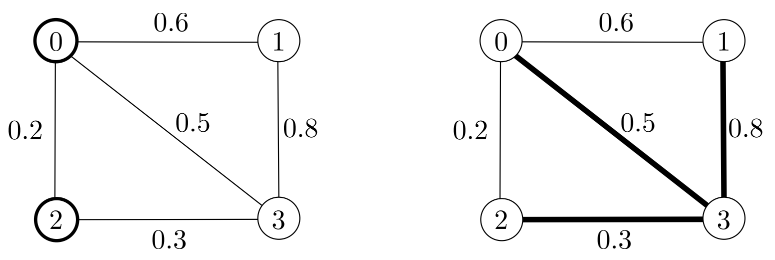

Figure 1: (left) Illustrating the graph described in Question 3(d), with $n = 4$, $m = 5$ and $A = \{0, 2\}$. (right) Showing an optimal solution.