CS141 Fall 1999

Midterm Exam, Friday 29th Oct, 1999

Lovingly prepared by:

Michalis Faloutsos

Name:

SSN (4 last digits):

You have to answer all problems. The points for each question are in brackets.

A good tactic is to read first all the questions and start from the questions you know best.

In all questions, explain your answers.

Use other paper as scrap-paper, but try to reply in the space below each question and the back of the same page. If additional space is needed, indicate it clearly on the given hand-out.

Only students that are registered for CS141 can take this exam.

Good Luck and...Don't Panic!

1. [**5**] Assume a recursive procedure that calls itself at least 1 and up to k times each time it runs as shown below in pseudocode. Note that i decreases by one in the child-procedures.

```
void foo(int i)
   if i > 0  {
   choose r randomly between [1 ... k]
   repeat r times
           do foo(i-1)
           end
  }
}
```

   (a) [**2**] Show what is the minimum number of procedure calls that the call foo(n) can create ($n > 0$).

   (b) [**3**] Show what is the maximum number of procedure calls that the call foo(n) can create ($n > 0$) .

   Hint: using induction may be a good idea.

2. **[25]** In the following graph, consider "a" as the source and execute

   (a) **[5]** Breadth First Search

   (b) **[10]** Prim's algorithm

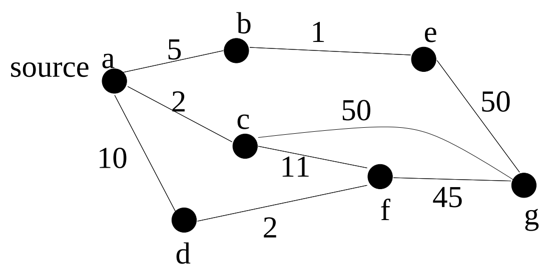
   (c) **[10]** Dijkstra's algorithm

   a) Show the order with which nodes are visited in BFS and the order with which they are selected (Prim - Dijkstra) and the value (key[] or d[]) with which they are selected, i.e., a(0), b(100), c(200) etc.

   Note 1: In case of a "tie" choose nodes in alphabetical order, i.e. b before c.

   Note 2: You don't have to produce the tables of the execution of the algorithm, but you can if you want to.

   b) Each algorithm creates a tree. Show all the intermediate trees that are being produced by each algorithm rooted at the source (i.e. all intermediate trees until algorithm stops).



3. **[10]** Assume a weighted undirected graph $G(V, E, w())$, where $w()$ is the weight function. We assume that all edge weights are **negative**, and a node s selected as source. We want to calculate the **maximum** weight paths from the source to all other nodes in the graph. Note that -5 is less than -2. Assume that we have implementations of all the algorithms we have seen in class that we will call "known" algorithms.

   (a) **[5]** Describe an algorithm that will calculate the max-weight paths using one or more known algorithms as a "black-box" i.e. not modifying their code.

   (b) **[5]** Given the correctness of the known algorithms, prove that your algorithm will calculate the correct max-weight paths.

4. [**15**]

(a) [**5**] A shortest path tree SP can be heavier than a minimum spanning tree MST in weighted undirected graphs. Give an example of a shortest paths tree for some root that is heavier than the MST for some graph. Show the graph, both trees and their cost.

(b) [**5**] Can a SP tree be lighter than an MST tree? If yes, show an example. If no, argue/prove that it can't.

(c) [**5**] Show that the ratio of the weights of the tree: $w(SP)/w(MST)$ can be as large as $O(N)$, where N the nodes in the graph and $w()$ the weight function. Hint: You may want to assume zero weight edges or edges with really really really small weight: $\epsilon \to 0$.

5. [**10**] Assume a weighted undirected graph $G(V, E, w())$ and assume that all the weights of the edges are distinct integers. Assume that we find a minimum spanning tree T. Prove that for every node $v \in V$, the minimum weight adjacent edge to that node belongs to the MST T, i.e.:

$$\forall v \in V, \quad e_v = \min_{w \in V} w(v, w) \quad and \quad e_v \in T$$

Hint: you may consider creating a cycle and then see if you can break it in a convenient way.