Problem Set 1

Labs: Feb 8-10, 2012
Due: Feb 16, 2012 10:30am

All problem sets must be handed in electronically by 10:30am on the due date.

Be sure to review the Electronic Submission document on the course website before handing in.
We will not grade submissions that do not meet these regulations.
Run `cs157_handin ps1` in the directory where your file is stored to hand in.

Please ensure that your solutions are complete, clear, and concise. Points will be deducted for overly complex solutions.

Problem 1

Consider the following high level algorithm that takes as input a weighted, undirected graph $G = (V, E)$ with n vertices and m edges.

REDUCE on graph $G = (V, E)$ such that all the weights in G are unique

$E_H \leftarrow \{(u, v) : v$ is the closest neighbor of u in $G\}$
$H \leftarrow (V, E_H)$
for each connected component c_i of H
  do add a corresponding vertex v_{c_i} to V'. It is associated to component c_i.
for each pair of connected components c_i, c_j of H
  do if G has edges between c_i and c_j
    then Add an edge between v_{c_i} and v_{c_j} to E' whose weight is
    the minimum weight of all edges between c_i and c_j in G.
return $G' = (V', E')$

1. Consider the following adjacency matrix for a weighted, undirected graph. An entry of 0 indicates no edge is present, while any other number signifies the weight of the edge.

	A	B	C	D	E	F	G	H
A	0	2	0	7	0	0	0	0
B	2	0	4	0	0	0	0	0
C	0	4	0	5	4	3	0	0
D	7	0	5	0	0	0	0	0
E	0	0	4	0	0	6	0	5
F	0	0	3	0	6	0	4	0
G	0	0	0	0	0	4	0	3
H	0	0	0	0	5	0	3	0

If you execute REDUCE on this graph, what is the output?

2. Prove that REDUCE can be implemented to run in $O(m)$ time.

3. Using the Reduce procedure and Prim's algorithm, design an algorithm that, on input graph G such that $m = O(n)$, finds the MST of G in $O(n \log \log n)$ time. You do not need to prove that your algorithm is correct.

4. Prove that your algorithm runs in $O(n \log \log n)$ time.

5. You can execute your algorithm even when m is larger than $O(n)$. Under what conditions on m and n is your algorithm faster than Prim's algorithm?

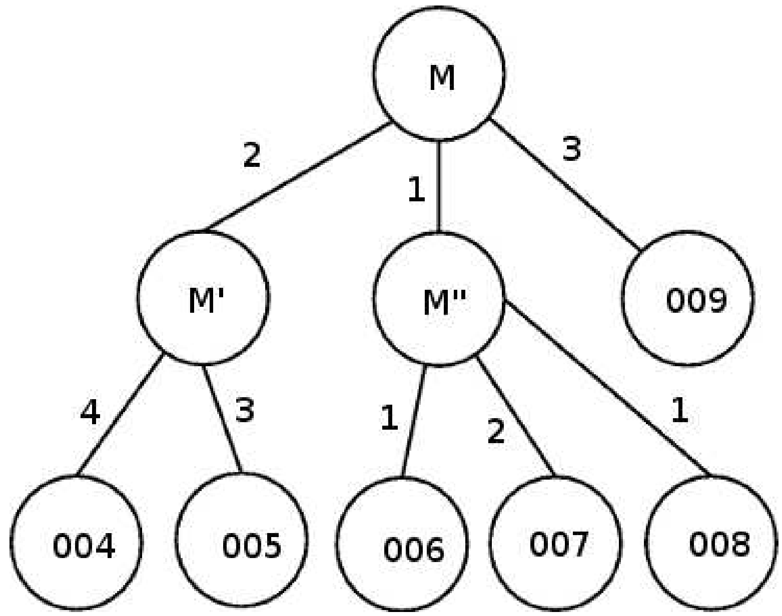

Figure 1: One of the British Secret Service Super-Secure Communication Networks.

Problem 2

You have just been hired by the British Secret Service (BSS) and your job is to synchronize the efforts of many **"00"** agents around the globe. Your communication network is a rooted tree $T = (V, E)$ comprised of vertices V and edges E with a root vertex $M \in V$ (See Figure 1 for an example). This network stretches vast distances and the communication time across each edge varies depending on the edge. Then the time for a message to each an agent x is the sum of communication times across all edges on the path from the root M to the leaf x. Your job is to send messages via this communication network so that all the messages are received by the agents simultaneously. Because the communication times vary from edge to edge, adding a delay d_e to some edges may be necessary to ensure that the agents receive messages at the same time (i.e. that the communication time from the root to each leaf is the same). However, the IT department at BSS does not like these delays. They have requested you add as few delays as possible (i.e. minimize $\sum_{e \in E} d_e$).

To summarize, given a tree T and communication times t_e, you must find an assignment of delays d_e minimizing $\sum_{e \in E} d_e$ and such that the communication times on all root-leaf paths are equal.

1. An assignment of d_e is said to be *feasible* if all the agents receive a message at the same time. Assuming the tree from Figure 1 with edge times as denoted, give two feasible delay assignments to the edges, one which is not optimal and one which is optimal.
2. Design an algorithm for assigning the optimal edge delays.
3. Prove the correctness of your algorithm.
4. Prove a running time of $O(n)$.

Problem 3

One of your friends, say his/her name is Alex, happens to be a teaching assistant for a course in the CS department. Once a week Alex prints out grades for the students in the sun lab. However, Alex always sleeps in, and a queue forms in the sun lab waiting for grades are printed one-by-one (as there is only one printer in the lab). Alex has noticed that some students chat more loudly while waiting for their grades and would like to reduce the disturbance in the lab. Your job, as Alex's loyal and brilliant friend, is to find what order Alex should print the grad sheets in so that the volume is as low as possible.

Alex has provided you with this useful information. The students are assigned anonymous identifiers from $1..n$, the volume of each student is v_i and the number of pages in that student's grade report is p_i (The TAs give **very** through feedback). Let C_i be the completion time for printing student i's grades. For example, if grade j was printed first then $C_j = p_j$, while if grade j came directly after grade i then $C_j = p_j + C_i$. In this model $\sum_{i=i}^{n} v_i C_i$ is a resonable approximation of the total room volume over time. Your task is to prioritize the list of students (i.e. find a permutation) such that the volume of the room over time is minimized if their grades are printed in that order.

i	p_i	v_i
1	7	2
2	5	7
3	1	4
4	3	15
5	10	1

Table 1: An Example of Alex's Useful Information.

1. Provide an optimal solution to the input data from Table 1. (You do not need to prove that it is optimal)
2. Design an algorithm for prioritizing grade printing.
3. Prove correctness.
4. Prove a running time of $O(n \log n)$.

Problem 4

1. For each of the following sets of numbers, state whether there exists an undirected graph $G = (V, E)$ whose node degrees are precisely those numbers. G should not contain multiple edges between the same pair of nodes or edges with both endpoints equal to the same node. All nodes must have positive degree. No proof is necessary.

   (a) 1, 1, 1, 2, 2, 2, 3, 3, 5

   (b) 1, 1, 2, 2, 2, 3

2. Give an example of d_1, d_2, d_3, d_4, d_5 such that all $d_i \leq 4$ and $d_1 + d_2 + d_3 + d_4 + d_5$ is even but for which no graph with degree sequence $(d_1, d_2, d_3, d_4, d_5)$ exists. No proof is necessary.

3. Come up with an algorithm that, given a list of n natural numbers $d_1, d_2, ..., d_n$, decides whether there exists an undirected graph $G = (V, E)$ whose node degrees are precisely $d_1, d_2, ..., d_n$, again with the stipulation that G should not contain multiple edges between the same pair of nodes or edges with both endpoints equal to the same node.

4. Prove the correctness of your algorithm.

5. Prove that your algorithm is linear in the sum of the node degrees.