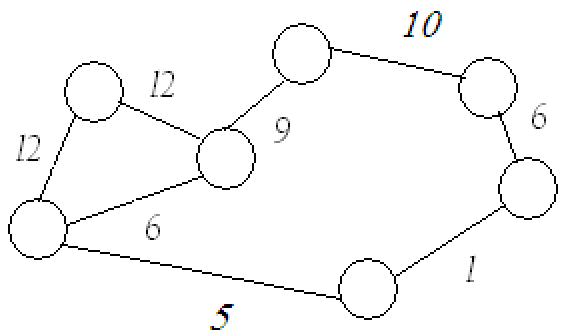
CS584
HW#3

DUE: Monday, October 19

For this homework, you should assume the existence of an algorithm to test if graph *G* is connected in time in $O(m) = O(|E|)$.

1. (15 points) For weighted graph $G = (V, E)$, let $\mu(G) = \max_{e \in E} \{w(e)\}$. If *G* is



then $\mu(G) = 12$.

a Describe an algorithm to accept as input a connected weighted graph *G* and yield as output a spanning subgraph *G*' of *G* to minimize $\mu(G')$. That is, *G*' should satisfy

$$\mu(G') = \min_{\text{spanning subgraph } H \text{ of } G} \{\mu(H)\}.$$

b Prove that your algorithm works correctly.

c Describe an algorithm which accepts as input connected weighted graph $G=(V, E)$ and $\omega \in \mathbb{R}^+$ and tests whether *G* admits a spanning tree *G*' such that $\mu(G') \le \omega$. Your algorithm should work in worst-case time in $O(m) = O(|E|)$.

2. (10 points) Given a connected weighted graph $G = (V, E)$ and edge $uv \in E$, describe an algorithm to test if there exists a MST of *G* which does not contain *uv*. The execution time of your algorithm should be in $O(n+m) = O(|V|+|E|)$. Prove that your algorithm works.

3. (8 points) Assume that a project consists of n jobs that must be executed in order, $(\pi_1, \ldots, \pi_n)$, and each job takes 1 unit of time. But the jobs must be run on one processor and they can not share time and their execution can not be pre-empted. The processor is available for times $t_1, t_2, \ldots$ on successive days. We want to finish the processing of the jobs in as few days as possible. For example, if n=8 and $t_1 = 2.3, t_2 = 3.2, t_3 = 4.2, t_4 = 4.4, \ldots$, then an optimal schedule (note that every schedule takes at least 3 days) is

DAY	JOBS
1	π_1
2	π_2, π_3, π_4
3	$\pi_5, \pi_6, \pi_7, \pi_8$

Describe an efficient algorithm to schedule the jobs using as few days as possible, and prove that your algorithm is optimal.

4. (6 points) Assume that we are implementing a binomial heap with eager UNION.

a. Describe a class of binomial heaps which cause EXTRACT-MIN to run in time $\Omega(\lg n)$.

b. Describe a class of binomial heaps which cause INSERT to run in time $\Omega(\lg n)$.

CS584
HW#3
SOLUTIONS

1. ***a*** A MST *G*' of *G* also minimizes $\mu(G')$ over all spanning trees of *G*. So Prim's algorithm or Kruskal's algorithm or any algorithm to compute a MST of *G* would solve our problem.

b Assume $G' = (V, E')$ is a MST of *G* but $G'' = (V, E'')$ is a spanning tree of *G* with $\mu(G') > \mu(G'')$. Choose some $e \in E'$ satisfying $w(e) > \mu(G'')$. There must be some edge *e*' from E'' in the principal cut of G' and *e*. Because $w(e) > \mu(G'')$, replacing *e* in E' yields a spanning tree of lower weight than *G*', contradicting that *G*' is a MST. Hence, *G*' must minimize $\mu(G')$ over all spanning trees of *G*.

c $E' \leftarrow \{e \in E | w(e) \le \omega\}$ $\quad O(m)$

Test if (V, E') is connected $\quad O(m)$

2. **for each** $e \in E$ **if** $w(e) > w(uv)$ **then** remove *e* from *E* $\quad O(m)$

$E \leftarrow E - \{uv\}$ $\quad O(1)$

if *u* and *v* belong to the same component of *G* $\quad O(m)$

then return "There exists an MST which doesn't contain *uv*" $\quad O(1)$

PROOF: If the new *E* spans a connected graph, then there exists a path from *u* to *v* of edges with weights $\le w(uv)$. Thus if we adds *uv* to this path, we can apply the Red Rule to eliminate *uv* from consideration for inclusion in an MST of *G*.

3. We use the following Greedy algorithm which packs as many jobs as possible into the next day.

$k \leftarrow 0$ /* *k* jobs have been scheduled */

$day \leftarrow 1$

repeat

schedule $\pi_{k+1}, \ldots, \pi_{k+\lfloor t_d \rfloor}$ for *day*

$k \leftarrow k + \lfloor t_d \rfloor$

$day \leftarrow day + 1$

until $k = n$

The algorithm works in time in $O(n)$, the time to read the input and the time to print the schedule, so the algorithm's exeution time is optimal.

CLAIM: The schedule produced by the algorithm is optimal.

PROOF: We show by induction that for each day, the algorithm runs as many jobs as possible by that day. Clearly at most $\lfloor t_1 \rfloor$ jobs can be run on day 1. Assuming that this is true for day d, and since at most $\lfloor t_{d+1} \rfloor$ jobs can be run on day d+1, the theorem is established.

4. ***a***. For any positive integer k, performing EXTRACT-MIN on a binomial heap of $n = 2^k - 1$ elements where the minimum element is the root of binomial tree $B_{k\text{-}1}$ exposes k-1 new binomial trees. A new set of k-1 links must be formed, in time $\Omega(\lg n)$.

b. For any positive integer k, performing INSERT on a binomial heap of $n = 2^k - 1$ elements links the new element with the roots of k-1 binomial trees, in time $\Omega(\lg n)$.