Algorithms and Complexity
Semester 2, 2012

Problem Set 3

J. Mestre

You should solve all problems above the line before the tutorial. Tutors will call out students to solve these problems in front of the class during the tutorials.

You should attempt to solve some of the problems below the line. Tutors will go over these problems during the tutorial.

Problem 1: Let $G = (V, E)$ be an undirected graph with edges weights $w : E \to R^+$. For all $e \in E$, define $w_1(e) = \alpha w(e)$ for some $\alpha > 0$, $w_2(e) = w(e) + \beta$ for some $\beta > 0$, and $w_3(e) = w(e)^2$.

1. Suppose p is a shortest s-t path for the weights w. Is p still optimal under w_1? What about under w_2? What about under w_3?

2. Suppose T is minimum weight spanning tree for the weights w. Is T still optimal under w_1? What about under w_2? What about under w_3?

Problem 2: It is not uncommon that a given optimization problem has multiple optimal solutions. For example, in an instance of the shortest s-t path problem, there could be multiple shortest paths connecting s and t. In such situations, it may be desirable to break ties in favor of a path that uses the fewest edges. Modify Dijkstra's algorithm so that the shortest paths found have fewest edges. You can assume that the edge lengths ℓ are positive integers.

1. Let us define a new edge function $\ell'(e) = M\ell(e)$ for each edge e. Show that if P and Q are two s-t paths such that $\ell(P) < \ell(Q)$ then $\ell'(Q) - \ell'(P) \geq M$.

2. Let us define a new edge function $\ell''(e) = M\ell(e) + 1$ for each edge e. Show that if P and Q are two s-t paths such that $\ell(P) = \ell(Q)$ but P uses fewer edges than Q then $\ell''(P) < \ell''(Q)$.

3. Show how to set M in the second function so that the shortest s-t path under ℓ'' is also shortest under ℓ and uses the fewest edges among all such shortest paths.

Problem 3: Suppose we are to schedule print jobs on a printer. Each job j has associated a weight $w_j > 0$ (representing how important the job is) and a processing time t_j (representing how long the job takes). A schedule is an ordering $\sigma = \langle j_1, j_2, \ldots, j_n \rangle$ of the jobs. Notice that the printer cannot start working on job j_i before finishing job j_{i-1}. For a given schedule σ, let $F(\sigma, j)$ be the finishing time of job j in the schedule.

Design a greedy algorithm that computes a schedule with minimum weighted average finishing time; in other words, we want to find a schedule σ minimizing $\sum_j w_j F(\sigma, j)$.

Problem 4: Consider the following generalization of the shortest path problem where in addition to edge lengths, each vertex has a cost. The objective is to find an s-t path

that minimizes the total length of the edges in the path plus the cost of the vertices in the path. Design an efficient algorithm for this problem.

Problem 5: Consider the following algorithm for the MST problem: tree:

```
IMPROVING-MST(G,w)

Let T be some spanning tree of G
For e ∈ E [in any order]
  T ← T + e
  Let C be the unique cycle in T
  Let f be a heaviest edge in C
  T ← T − f
return T
```

Prove its correctness and analyze its time complexity. To simplify things, you can assume the weights are different.

Problem 6: A computer network can be modelled as an undirected graph $G = (V, E)$ where vertices correspond to computers and edges correspond to physical links between computers. In practice the maximum transmission rate or *bandwidth* varies from link to link; let $b(e)$ be the bandwidth of edge $e \in E$. The bandwidth of a path P is defined as the minimum bandwidth of edges in the path, i.e., $b(P) = \min_{e \in P} b(e)$.

Suppose we number the vertices in $V = \{v_1, v_2, \ldots, v_n\}$. Let $B \in R^{n \times n}$ a matrix where B_{ij} is the maximum bandwidth between v_i and v_j. Give an algorithm for computing B. Your algorithm should run in $O(n^3)$ time.

Problem 7 (Advanced): Solve the previous problem in $O(n^2)$ time.