ORIE 6330 Network Flows September 13, 2012

Problem Set 2

Due Date: September 27, 2012

1. In class, we discussed some tricks for speeding up the push/relabel algorithm in practice. Here is another such trick, called *gap relabeling*. Suppose there is a value $k < n$ such that there are no nodes i with $d(i) = k$, but there are active nodes j of distance $k < d(j) < n$. Prove that setting $d(j) = n$ for all such nodes gives a valid distance labeling. Note that if we use the variant of the algorithm in which a node is active only if $e_f(i) > 0$ and $d(i) < n$, then this relabeling can reduce the number of active nodes.

2. Another variant of the push-relabel algorithm is called *FIFO* push-relabel. This version maintains a queue of active nodes; initially all active nodes are added to the queue. The algorithm takes a node i from the front of the queue and keeps performing push and relabel operations to i until there is no longer any excess at i. If pushing flow from i to j causes j to become active, and j is not already in the queue, the algorithm adds j to the end of the queue.

   To bound the running time of the algorithm, we need to bound the number of nonsaturating pushes performed by the algorithm. To do this, we use a potential function argument on *passes* over the queue. The first pass over the queue ends when the algorithm has performed a discharge operation on all the nodes initially added to the queue. In general, the kth pass to the queue ends when the algorithm has performed a discharge operation on all the nodes added to the queue in the $(k-1)$st pass. Consider the potential function $\Phi = \max_{\text{active } i} d(i)$.

   (a) Use the potential function to prove that the algorithm makes $O(n^2)$ passes.

   (b) Argue that the bound on the number of passes implies there are $O(n^3)$ nonsaturating pushes.

   (c) Finally, argue that the FIFO version of push-relabel takes $O(n^3)$ time.

3. In a variation of the normal maximum flow problem, we have a *parametric* network, in which the capacities of arcs leaving the source and entering the sink vary with a parameter λ. Let $u(i, j, \lambda)$ be the capacity of arc (i, j) for parameter λ. In particular, we have

   - $u(s, j, \lambda)$ is a nondecreasing function of λ for all $j \neq t$;
   - $u(i, t, \lambda)$ is a nonincreasing function of λ for all $i \neq s$;
   - $u(i, j, \lambda) = u(i, j)$ for all $i \neq s$ and $j \neq t$.

   In the parametric max flow problem, in addition to the usual input for the maximum flow problem, we are also given the values $\lambda_1 < \lambda_2 < \cdots < \lambda_\ell$, and the capacities of the arcs $u(i, j, \lambda_k)$ for all $(i, j) \in A$, $1 \leq k \leq \ell$. The goal is to find flow values

$f_1, \ldots, f_\ell$ and minimum s-t cuts $S_1, \ldots, S_\ell$ for the flow problems associated with the capacities given by the input $\lambda_1, \ldots, \lambda_k$.

(a) Show that the push/relabel algorithm for the maximum flow problem can be used to solve the parametric maximum flow problem in $O(n^2(\ell + m))$ time. (Hint: Start by solving the flow problem for λ_1. What should you do after that?)

(b) Show that $S_1 \subseteq S_2 \subseteq \cdots \subseteq S_\ell$.

(c) Show that there are at most $n - 1$ distinct sets among the S_k.

4. Consider the problem of finding shortest paths in a directed graph:

Shortest paths in directed graphs

- **Input:**
  - A directed graph $G = (V, A)$
  - Lengths $l(i, j)$ for all $(i, j) \in A$, integer, possibly negative; however, no negative length cycles
  - Source vertex $s \in V$.
- **Goal:** Find length $d(v)$ of shortest path from s to v for all $v \in V$.

The Bellman-Ford shortest path algorithm computes $d(i)$ by computing $d_k(i)$, the shortest walk (i.e. a path in which vertices can be repeated) between s and i using exactly k arcs.

(a) Prove that $d_k(j)$ can be computed by the recurrence

$$d_k(j) = \min_{(i,j) \in A} [d_{k-1}(i) + l(i, j)].$$

(b) Let $h_l(i) = \min_{k=1,\ldots,l} d_k(i)$. Prove that if the graph has no negative-length cycle then $h_{n-1}(i) = d(i)$ for all $i \in V$. Moreover, show that the graph has no negative-length cycle iff for all $i \in V$, $d_n(i) \geq h_{n-1}(i)$. (Hint for the if part: prove that if $h_n(i) = h_{n-1}(i)$ for all i then $h_c(i) = h_n(i)$ for all $c \geq n$.)

(c) Prove that this algorithm runs in $O(mn)$ time.