CS 573: Algorithms, Fall 2009
Homework 3, due Wednesday, October 28, 23:59:59, 2009

Version 1.0

Name:	
Net ID:	Alias:

#	Score	Grader
1.		
2.		
3.		
4.		
Total		

Neatly print your name(s), NetID(s), and the alias(es) you used for Homework 0 in the boxes above. Staple this sheet to the top of your homework. If you are on campus, submit the homework by submitting it in SC 3306 (or sliding it under the door).

> At other times you seemed to me either pitiable or contemptible, eunuchs, artificially confined to an eternal childhood, childlike and childish in your cool, tightly fenced, neatly tidied playground and kindergarten, where every nose is carefully wiped and every troublesome emotion is soothed, every dangerous thought repressed, where everyone plays nice, safe, bloodless games for a lifetime and every jagged stirring of life, every strong feeling, every genuine passion, every rapture is promptly checked, deflected and neutralized by meditation therapy.
>
> – The Glass Bead Game, Hermann Hesse

Required Problems

1. PROVE INFEASIBILITY.
   [25 Points]

   You are trying to solve a circulation problem, but it is not feasible. The problem has demands, but no capacity limits on the edges. More formally, there is a graph $G = (V, E)$, and demands d_v for each node v (satisfying $\sum_{v \in V} d_v = 0$), and the problem is to decide if there is a flow f such that $f(e) \geq 0$ and $f^{in}(v) - f^{out}(v) = d_v$ for all nodes $v \in V$. Note that this problem can be solved via the circulation algorithm from Section 7.7 by setting $c_e = +\infty$ for all edges $e \in E$. (Alternately, it is enough to set c_e to be an extremely large number for each edge – say, larger than the total of all positive demands d_v in the graph.)

   You want to fix up the graph to make the problem feasible, so it would be very useful to know why the problem is not feasible as it stands now. On a closer look, you see that there is a subset U of nodes such that there is no edge into U, and yet $\sum_{v \in U} d_v > 0$. You quickly realize that the existence of such a set immediately implies that the flow cannot exist: The

set U has a positive total demand, and so needs incoming flow, and yet U has no edges into it. In trying to evaluate how far the problem is from being solvable, you wonder how big the demand of a set with no incoming edges can be.

Give a polynomial-time algorithm to find a subset $S \subset V$ of nodes such that there is no edge into S and for which $\sum_{v \in S} d_v$ is as large as possible subject to this condition.

2. Even More Capacitation.
[25 Points]

In a standard $s - t$ Maximum-Flow Problem, we assume edges have capacities, and there is no limit on how much flow is allowed to pass through a node. In this problem, we consider the variant of the Maximum-Flow and Minimum-Cut problems with node capacities.

Let $G = (V, E)$ be a directed graph, with source $s \in V$, sink $t \in V$, and nonnegative node capacities $\{c_v \geq 0\}$ for each $v \in V$. Given a flow f in this graph, the flow through a node v is defined as $f^{in}(v)$. We say that a flow is feasible if it satisfies the usual flow-conservation constraints and the node-capacity constraints: $f^{in}(v) \leq c_v$ for all nodes.

Give a polynomial-time algorithm to find an s-t maximum flow in such a node-capacitated network. Define an s-t cut for node-capacitated networks, and show that the analogue of the Max-Flow Min-Cut Theorem holds true.

3. Maximum Flow By Scaling
[25 Points]

Let $G = (V, E)$ be a flow network with source s, sink t, and an integer capacity $c(u, v)$ on each edge $(u, v) \in E$. Let $C = max_{(u,v) \in E} c(u, v)$.

(a) **[2 Points]** Argue that a minimum cut of G has capacity at most $C|E|$.

(b) **[6 Points]** For a given number K, show that an augmenting path of capacity at least K can be found in $O(E)$ time, if such a path exists.

The following modification of Ford-Fulkerson-Method can be used to compute a maximum flow in G.

```
Max-Flow-By-Scaling(G, s, t)
1    C ← max_{(u,v)∈E} c(u, v)
2    initialize flow f to 0
3    K ← 2^⌊lg C⌋
4    while K ≥ 1 do {
5         while (there exists an augmenting path p of
                        capacity at least K) do {
6              augment flow f along p
          }
7         K ← K/2
     }

8    return f
```

(c) **[4 Points]** Argue that Max-Flow-By-Scaling returns a maximum flow.

(d) **[5 Points]** Show that the capacity of a minimum cut of the residual graph G_f is at most $2K|E|$ each time line 4 is executed.

(e) **[5 Points]** Argue that the inner **while** loop of lines 5-6 is executed $O(E)$ times for each value of K.

(f) **[3 Points]** Conclude that MAX-FLOW-BY-SCALING can be implemented so that it runs in $O(E^2 \lg C)$ time.

4. NUMBER OF AUGMENTING PATHS
[25 Points]

(a) **[13 Points]** Show that a maximum flow in a network $G = (V, E)$ can always be found by a sequence of at most $|E|$ augmenting paths. [Hint: Determine the paths after finding the maximum flow.]

(b) **[12 Points]** Suppose that a flow network $G = (V, E)$ has symmetric edges, that is, $(u, v) \in E$ if and only $(v, u) \in E$. Show that the Edmonds-Karp algorithm terminates after at most $|V||E|/4$ iterations. [Hint: For any edge (u,v), consider how both $\delta(s, u)$ and $\delta(v, t)$ change between times at which (u,v) is critical.]