Assignment sheet 1

Due Date: 1 February 2006

---

1. Show the following flow decomposition theorem:

   Any $s-t$ flow f can be decomposed into at most m "primitive elements", where a primitive element is:

   (i) a path P from s to t where the flow on each edge of the path is $\delta(P)$, or

   (ii) a simple cycle Γ where the flow on each edge of the cycle in $\delta(\Gamma)$.

   [Note that the decomposition in not unique. The various paths and cycles are not necessarily edge disjoint - i.e., an edge can be a member of multiple primitive elements. The flow on an edge includes the flow of all paths and cycles that include the edge.]

2. Consider the following algorithm for computing max flow in a directed graph G with integral capacities $\leq C$.

   - Assume that we represent capacities using k bits, where $k = \lfloor log\ C \rfloor + 1$.

   - Denote by G^i the network derived form G by looking at the i most significant bits of each capacity.

   1. Find max flow f in G^1.
   2. $i = 2$.
   3. Repeat
      - look at G^i with flow $2f$.
      - find max flow in residual network: G^i_{2f}.
      - $f = 2f +$ max flow in G^i_{2f}.
      - $i = i + 1$.

   until $i = k + 1$.

   Show that the above algorithm correctly computes a max flow in G. Show that its running time is $O(m^2 log\ C)$ where m is the number of edges in G.

   (Hint: For the running time estimate, show that every $s-t$ path in G^{i+1}_{2f}, where f is a max flow in G^i, has $\delta = 1$. Then use the fact that every max-flow can be decomposed into at most m $s-t$ paths.)

3. The Min-Flow problem:

   Assume that the capacity constraints involve lower bounds also, that is we have: $l(e) \leq f(e) \leq c(e)\ \forall e \in G$. Now consider the opposite of the max-flow problem, which is to *minimize* the net flow into t under the capacity constraints and flow conversation constraints.

- Show how to solve this problem by first finding a feasible solution, and by then using a max-flow algorithm.
- Derive an analog to the max-flow min-cut theorem.

4. The *edge connectivity* of an undirected graph is the minimum number k of edges that must be removed to disconnect the graph. For example, the edge connectivity of a tree is 1, and the edge connectivity of a simple cycle is 2. Show how the edge connectivity of an undirected graph $G = (V, E)$ can be determined by running a max-flow algorithm on at most $|V|$ flow networks.

5. Prove the following theorem of Hall using the max-flow min-cut theorem.

   A bipartite graph $G = (U \cup V, E)$ has a U-perfect matching (= a matching of size $|U|$) iff for every subset $U' \subset U$, we have $|N(U')| \geq |U'|$.

   [$N(U')$ is the set of neighbours of the vertices in U'.]

6. Consider the problem of placing n queens on a chessboard of dimension $n \times n$ so that there is no pair of queens that attack each other.

   - Formulate the problem of finding a solution as a max-flow problem.
   - Formulate the problem of counting the number of distinct solutions as a max-flow problem.