Fall 2007 **CMSC 858K: Homework 4** Samir Khuller

Due in class: Oct 11.

Problems are to be done by yourself (unlesss specified) with no help from another person. If you use any sources other than the textbooks for the class, you are to cite them. In any case, the writeup should be your own and not simply taken from another source.

(1) Design a polynomial time algorithm to find a minimum weight vertex cover in a bipartite graph by directly reducing it to network flow. Nodes have weights, and we wish to find a minimum weight subset of nodes that cover all edges.

(2) (Approach proposed by Matt McCutchen.) The following modified version of the Bellman–Ford shortest-path algorithm finds a negative-length cycle in a graph $G = (V, E)$ if one exists; if not, it finds a zero-length cycle if one exists. The binary search for a min-mean cycle can use this algorithm as a subroutine, although Martin's idea to remove the need to find zero-length cycles by remembering the cycle that gave the right endpoint of the search interval is even better.

For each vertex v, the algorithm stores both $d[v]$, the shortest distance to v found so far, and $k[v]$, the greatest number of edges in any path giving that shortest distance.

Initialize both $d[v]$ and $k[v]$ to zero for every vertex. To relax along an edge of length l from u to v, do the following:

- If $d[u] + l < d[v]$, set $d[v] \leftarrow d[u] + l$ and $k[v] \leftarrow k[u] + 1$.
- Otherwise, if $d[u] + l = d[v]$ and $k[u] + 1 > k[v]$, set $k[v] \leftarrow k[u] + 1$.

If all the labels stabilize within $|V|$ passes, report that neither a negative-length nor a zero-length cycle exists. If the labels do not stabilize, review the label changes made on the last pass. If at least one $d[v]$ value changed, choose a vertex v such that $d[v]$ changed, follow parent pointers from v until you enters a cycle, and report the cycle as negative-length. Otherwise, only $k[v]$ values changed; choose a vertex v such that $k[v]$ changed, trace back to a cycle, and report the cycle as zero-length.

Prove the correctness of this method.

(3) Suppose a company has the following vehicle requirements. Jan: 230 cars, Feb: 210 cars, Mar: 440 cars, Apr: 390 cars, May: 425 cars, Jun: 450 cars. They have the following lease options. A 3 month lease for 1700$, a 4 month lease for 2200$, a 5 month lease for 2600$.

Show how to formulate this problem as a linear (integer) programming problem, and then argue that this can be reduced to a min cost flow computation.

(4) We have a periodic scheduling problem. There are n tasks. Each task takes one unit of time to perform. The requirement is that task i should be scheduled once in each time period p_i.

For example if task i, has period p_i. We must schedule it in each time-window of the form $[p_i \cdot j + 1, \ldots, p_i \cdot (j + 1)]$ for $j = 0 \ldots$. Let $L = lcm(p_1, p_2, \ldots, p_n)$. Moreover suppose that $\sum_i \frac{1}{p_i} \leq 1$. Show how to find a periodic schedule for the first L time slots. If the above condition is not satisfied, there is no periodic schedule. Why?

(5) Given an undirected unweighted graph G, we wish to compute a subgraph in which every vertex has degree at most 2. Moreover, we wish to maximize the number of edges in the subgraph. How can we reduce this to matching?