Fall 2001 **CMSC 451: Project Part 1** Samir Khuller

Due in class: Nov 27.

This is an outline of two approaches. If this outline is not clear, please talk to me or the TA.

If we assume that the costs are postive integers, then there is a simple dynamic programming solution. Define a function $f(v, c)$ that denotes the length of the shortest path from s to v with cost at most c. Our goal is to compute $f(t, C)$ which denotes the shortest path from s to t with cost at most C. If no path with cost at most c exists then the f value is ∞.

Let $IN(v)$ denote the set of vertices that have edges to v. We can compute a table of size $C \cdot |V|$ in increasing row order (each row corresponds to a cost value). If each edge has cost > 0 then this is an efficient algorithm.

$$f(c, v) = \min_{u \in IN(v) | c(u,v) \leq c} f(u, c - c(u, v)) + l(u, v)$$

The running time should be $O(EC)$.

If we wish to allow for zero cost edges...the following (slower) algorithm will work. For each vertex v we maintain a list of values (i, l_i) which denotes the shortest length path from s to v with cost at most i. (We can do this for each value for $i = 1 \ldots C$.)

Define the operation $RELAX(u, v)$ as follows.

```
For each $i$ in $U(i,l_i)$ do
Let $j= i+c(u,v)$
  If $l_i + l(u,v) < l_j$ then $l_j = l_i + l(u,v)$.
```

The proof should be almost the same as the Bellman-Ford Algorithm, where after we perform the RELAX operation on all the edges in k rounds, we can claim that the optimal path of a certain cost that has k edges has been computed correctly. The running time should be $O(EVC)$.