600.363/463 Solution to Homework 7
Posted: 10/29/03
Due: Wednesday, 11/5/03

24.1-3

The proof of Lemma 24.2 shows that for every v, $d[v]$ has attained its final value after *length* (any shortest-weight path to v) iterations of BELLMAN-FORD. Thus after m passes, BELLMAN-FORD can terminate. We don't know m in advance, so we can't make the algorithm loop exactly m times and then terminate. But if we just make the algorithm stop when nothing changes any more, it will stop after $m + 1$ iteration (i.e., after one iteration that makes no changes.).

BELLMAN-FORD$-(m + 1))(G, w, s)$
INITIALIZE-SINGLE-SOURCE(G, s)
changes $\leftarrow$ true
while changes=true
    do changes $\leftarrow$ false
        for each edge$(u, v) \in E[G]$
            do RELAX-M(u, v, w)

RELAX-M(u, v, w)
    if $d[v] > d[u] + w(u, v)$
        then $d[v] \leftarrow d[u] + w(u, v)$
            $\pi[u] \leftarrow u$
            changes $\leftarrow$ true

24.1-5

We can modify the INITIALIZE-SINGLE-SOURCE(G, s) as:

for each vertex $V \in V[G]$
    do $d[v] \leftarrow \infty$
        $\pi[v] \leftarrow NIL$
for each edge $(u, v) \in E[G]$
    if $d[v] > w(u, v)$
        $d[v] \leftarrow w(u, v)$

Then we execute BELLMAN-FORD algorithm and get the correct result. Since we initialize $d[v]$ as the weight of shortest path to v from some vertex u, and during the RELAX, we have $d[v]$ as current minimum distance from some vertex u in V to v. After $||V|| - 1$ iterations, each v has $d[v] = \delta^*(v)$ since the longest path is $||V|| - 1$.

24.3-4

Solution 1: To find the most reliable path between s and t, run Dijkstra's algorithm with edge weights $w(u, v) = -lgr(u, v)$ to find the shortest paths from s in $O(E + V \lg V$ time. The most reliable path is the shortest path from s to t, and that path's reliability is the product of the reliabilities of its edges.
Explanation:
Because the probabilities are independent, the probability that a path will not fail is the product of the probabilities that its edges will not fail. We want to find a path $s \to t$ since that $\prod_{(u,v)\in p} r(u, v)$ is maximized. This is equivalent to maximizing $\lg(\prod_{(u,v)\in p} r(u, v) = \sum_{(u,v)\in p} \lg r(u, v)$, which is equivalent to minimizing $\sum_{(u,v)\in p} - \lg r(u, v)$. (Note $r(u, v)$ can be 0, and $\lg 0$ is underfined. So in this algorithm, define $\lg 0 = -\infty$.) Thus if we assign weights $w(u, v) = -\lg r(u, v)$, we have a shortest-path problem.
Since $\lg 1 = 0, \lg x < 0 for 0 < x < 1$, and we have defined $\lg 0 = -\infty$, all the weights w are nonnegative, and we can use Dijkstra's algorithm to find the shortest paths from s in $O(E + V \lg V)$ time.

Solution 2 You can also work with the original probability by running a modified version of Dijkstra's algorithm that maximizes the product of reliabilities along a path instead of minimizing the sum of weights along a path.
In Dijkstra's algorithm, use the reliabilities as edge weights and substitute:
1. max (and Extract-Max) for min (and Extract-min) in relaxation and queue.
2. $\times$ for $+$ in relaxation
3. 1 (identity for $\times$) for 0 (identity for $+$) and $-\infty$ (identity for min) for ∞ (identity for max).
FOr example, the following is used for instead of the usual RELAX procedure:

RELAX-RELIABILITY(u, v, r)
1 if $d[v] < d[u] \cdot r(u, v)$
2   then $d[v] \leftarrow d[u] \cdot r(u, v)$
3         $\pi[v] \leftarrow u$

This algorithm is isomorphic to the one above: it performs the same operations except that it is working with the original probabilities instead of the transformed ones.