Homework 10 Solutions

Note: The correctness of the algorithms has not been analyzed extensively, for brevity purposes. If you have any question, please contact the instructor or the TA.

**Problem 1 Solution:** Initially, the algorithm topologically sorts the DAG and produces a linear ordering on the vertices. This is performed in \( \Theta(V + E) \) time. Then, we make one pass over the vertices in the topologically sorted order. As each vertex is processed, all the edges that leave the vertex are relaxed. Here is the pseudocode.

```plaintext
run topological_sort(G)

/* Init() */

for each vertex \( v \) in \( V \)
  \( d[v] = -\infty; \) /* \( \infty \) denotes infinity */

\( d[s] = 0; \) /* \( s \) is the source */

/* Update */

for each vertex \( u \) in topologically sorted order
  for each vertex \( v \) in \( \text{Adj}[u] \)
    if \( d[v] < d[u] + \text{w}(u, v) \)
      \( d[v] = d[u] + \text{w}(u, v); \)

This takes time \( O(V + E) \).

Correctness: We must show that at the termination of the algorithm, the maximum weighted path is computed from \( s \) to every destination \( v \). Let \( p(v, u) \) be the maximum path weight from \( v \) to \( u \). If \( u \) is not reachable from \( s \), then \( d[v] = p(s, v) = -\infty \). If \( v \) is reachable from the source \( s \), there is a maximum weighted path \( u = < u_0, u_1, \ldots, u_k > \), where \( v_0 = s \) and \( v_k = v \). Because of the topological sort, the edges on the path are relaxed in the order \((u_0, u_1), (u_1, u_2), \ldots, (u_{k-1}, u_k)\). Using induction as in the proof of correctness for Bellman-Ford (taught in the class) it can be proved that \( d[v_i] = p(s, v_i) \) at termination for \( i = 0, 1, \ldots, k \).
Problem 2 Solution: The Bellman-Ford algorithm will be used for the detection of negative weight cycle. It will return a boolean value which will indicate whether there is a negative weight cycle or not in the strongly connected graph.

The algorithm uses the same basic pseudo-code taught in class for Bellman-Ford. It actually enhances this code, by adding the following step in the previous code:

/* t here equals to V */
for every vertex v in V
    for every vertex u in Adj[v]
        if d_{t}(v) > d_{t-1}(u) + w(u, v)
            return false

return true

The existing code of Bellman-Ford costs $O(VE)$. This step costs $O(E)$, so total complexity is $O(VE)$.

Correctness: We have to prove that if the graph contains a negative weight cycle, then the algorithm will return false.

Let $c = <v_0, v_1, \ldots, v_k>$ where $v_0 = v_k$, be a negative weight cycle. This means,

$$\sum_{i=1}^{k} w(v_{i-1}, v_i) < 0$$

Assume that the algorithm does not return true, so that $d[v_i] \leq d[v_{i-1}] + w(v_{i-1}, v_i)$ for $i = 1, \ldots, k$.

Using the above inequality,

$$\sum_{i=1}^{k} d[v_i] \leq \sum_{i=1}^{k} d[v_{i-1}] + \sum_{i=1}^{k} w(v_{i-1}, v_i) \quad (1)$$

Since the graph is strongly-connected, $d[v_1]$ is finite. Also, $\sum_{i=1}^{k} d[v_i] = \sum_{i=1}^{k} d[v_{i-1}]$ (2). So, (1),(2) conclude

$$0 \leq \sum_{i=1}^{k} w(v_{i-1}, v_i)$$

which is a contradiction. QED
Problem 3 Solution: Consider the following counter-example. The graph has 3 vertices $s$, $a$, $b$ and edges $(s,a)$, $(s,b)$, $(b,a)$ with weights $1$, $2$, $-3$ respectively. Dijkstra will conclude that the shortest path weight from $s$ to $a$ is $1$. But the actual shortest path weight is $-1$, following the edges $(s,b)$, $(b,a)$. QED

Problem 4 Solution: A modification of Bellman-Ford is proposed. Initially, set all $d[v]=0$. Then, the relaxation is modified as follows:

$$d_t(v) = \min( d_{t-1}[v], \min_{w \in \text{Adj}(u)} (d_{t-1}(u) + w(u,v)) )$$

The algorithm has the same complexity as Bellman-Ford, that is $O(VE)$.

Its correctness depends on the correctness of Bellman-Ford. In case that all edges of the graph have positive weight, then the initialization $d[v]=0$ will remain unchanged during the algorithm: the minimum shortest path for every vertex is the one from itself. In case that there exist negative weight edges which make a shortest path to be negative, then the update will take place.