

TUTORIAL SHEET 14

1. Recall Dijkstra's algorithm discussed in class. Let all edges have positive length and s be the source vertex. Let v_1, \dots, v_n be the vertices in increasing order of shortest path distance from s (of course, the algorithm needs to figure out this ordering). Clearly $v_1 = s$. Now suppose we know v_1, \dots, v_i and let S denote the set $\{v_1, \dots, v_i\}$. Suppose we also know $d[v]$, the shortest path distance from s to v for each vertex $v \in S$.
 - For each vertex $w \notin S$, let $D[w]$ denote $\min_{(v,w) \in E, v \in S} (d[v] + l(v, w))$. Show that $D[w]$ is at least the length of the shortest path from s to w .
 - Let $x \notin S$ be the vertex with the minimum $D[x]$ value. Show that $D[x]$ is equal to the length of the shortest path from s to x .
2. Let $G = (V, E)$ be a directed graph and $s \in V$ be a designated source vertex. Assume that each edge $(u, v) \in E$ has a real-valued length $l(u, v)$ which may be positive or negative, but there are no negative-length cycles in G . Let $d[v]$ denote the length of the shortest path from s to v in G .

Consider the following algorithm:

Algorithm Bellman-Ford(G, s):

for each vertex v in V :

$D[v] := \text{infinity}$

$D[s] := 0$

for $i = 1$ to $|V| - 1$:

for each edge (u, v) in E :

if $D[u] + l(u, v) < D[v]$:

$D[v] := D[u] + l(u, v)$

- Show that for any vertex v , $D[v]$ remains at least $d[v]$ throughout the algorithm.
 - Prove that after the k -th iteration of the outer loop, the value $D[v]$ equals the length of the shortest $s \rightarrow v$ path that uses at most k edges.
 - Suppose we modify the algorithm to run the outer loop only $\lceil |V|/2 \rceil$ times. Construct an example graph (with edge lengths, possibly negative) for which this modified version fails to compute correct shortest path distances.
3. You are given a DAG and a source vertex s . Show how to efficiently compute for each vertex v , the number of distinct paths from s to v .
 4. You are given a directed graph where the edge lengths are positive. Show how to efficiently compute for each vertex v , the number of distinct shortest paths from s to v .