CS 344: Floyd-Warshall ALGORITHM

Assume $G = (V, E)$ is a directed graph with edge weights that are nonnegative. The Floyd-Warshall algorithm computes the shortest paths between all pairs.

For simplicity denote V by $\{1, \ldots, n\}$. Let $W = (w_{ij})$ be the matrix of the edge weights. $w_{ij} = \infty$ if the edge (i, j) does not exist.

Define $p_{ij}^{(k)}$ to be the shortest path from i to j using only vertices $1, \ldots, k$.

It is easy to see that if k does not appear on the shortest path from i to j using only $\{1, \ldots, k\}$, then

$$p_{ij}^{(k)} = p_{ij}^{(k-1)}.$$

If k does appear on the shortest path from i to j using only $\{1, \ldots, k\}$, then

$$p_{ij}^{(k)} = p_{ik}^{(k-1)} + p_{kj}^{(k-1)}.$$

Using this, we define $p_{ij}^{(k)}$ recursively. For $k = 0$,

$$p_{ij}^{(0)} = w_{ij}.$$

Next for $k = 1, \ldots, n$ we set

$$p_{ij}^{(k)} = \min\{p_{ij}^{(k-1)}, p_{ik}^{(k-1)} + p_{kj}^{(k-1)}\}.$$

The algorithm uses the matrix $W = (w_{ij})$ and the matrix D which will give the length of shortest paths.

FLOYD-WARSHALL (W)
$D \leftarrow W$
For $k \leftarrow 1$ to n
  For $i \leftarrow 1$ to n
    For $j \leftarrow 1$ to n
      $d_{ij}^{(k)} = \min\{d_{ij}^{(k-1)}, d_{ik}^{(k-1)} + d_{kj}^{(k-1)}\}.$
return $D = (d_{ij}^{(n)})$

Constructing Shortest Paths.

Define the predecessor array $\Pi = (\pi_{ij})$ as follows:
If $i = j$, or $w_{ij} = \infty$, then set $\pi_{ij}^{(0)} = Nil$. Otherwise, set $\pi_{ij}^{(0)} = i$.
For $k \geq 1$, define $\pi_{ij}^{(k)}$ as follows:
If $d_{ij}^{(k)} \leq d_{ik}^{(k-1)} + d_{kj}^{(k-1)}$ then set $\pi_{ij}^{(k)} = \pi_{ij}^{(k-1)}$. Otherwise, set $\pi_{ij}^{(k)} = \pi_{kj}^{(k-1)}$.

The complexity of the algorithm is $O(n^3)$.