Artificial Intelligence & Machine Learning

Bodhayan Roy

Department of Mathematics,
Indian Institute of Technology Kharagpur

Lecture 3

Dijkstra

1. Mark all vertices as unvisited. Initialize distance d_v of all vertices v from source as 0. Mark the source as an active vertex.
2. Do unless all vertices are marked depleted: Find the lowest weighted edge (u, v) such that u is active and v is unvisited. Set $d_v = d_u + w$ where w is the weight of (u, v). If all neighbours of u are active or depleted, then mark u as depleted.

Floyd Warshall

1. Create an $n \times n$ matrix M where $M[i][j]$ will eventually store the length of a shortest path between vertices i and j. Initialize the diagonal elements of M to 0. Initialize all other elements of M to infinity.
2. If the edge (i, j) exists, then $M[i][j] = w$, where w is the weight of (i, j).
3. $\forall k \forall i \forall j$, do: If $M[i][j] > M[i][k] + M[k][j]$, then $M[i][j] = M[i][k] + M[k][j]$.

Bellman Ford

1. For each vertex v, assign to it the distance d_v from the source vertex. Initialize d_v to infinity for all i, except for the source vertex s for which $d_s = 0$.
2. Repeat $|V| - 1$ times: For every edge (u, v) (say, with weight w), if $d_u + w < d_v$ then $d_v = d_u + w$.
3. If for any edge (u, v), $d_u + w < d_v$, then report the existence of a negative cycle. Else each d_v gives the distance of v from the source. Path can be generated by assigning a neighbour u as predecessor of v such that $d_u + w = d_v$.