Lec 13-Graph Algorithms

All pair shortest path algorithm

- The **all-pairs shortest path problem** is the determination of the **shortest graph distances** between **every pair of vertices** in a given graph.

Given a weighted digraph $G = (V, E)$ with a weight function $w : E \to \mathbf{R}$, where R is the set of real numbers, determine the length of the shortest path (i.e., distance) between all pairs of vertices in G.

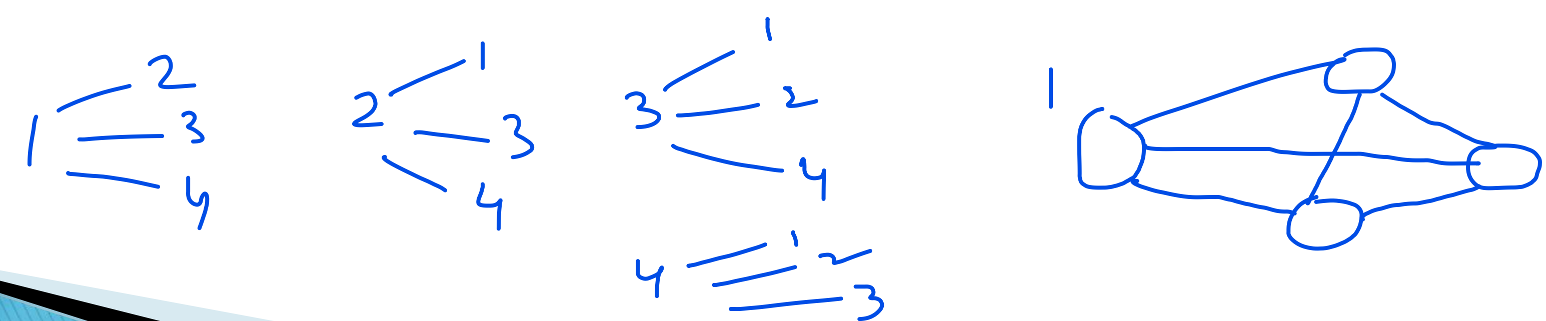
assume that there are no cycle with zero or negative cost.

All pair shortest path algorithm

Solution-1

- Assume no negative edges.
  - Run Dijkstra's algorithm, V times, once with each vertex as source with more sophisticated data structures.

- Complexity:
  - Around O(V * (V + E)Log V))
  - In worst case can go upto O(V*V*V Log V) times
  -



All pair shortest path algorithm

Solution-2

- Assume no negative weight cycles.
  - Run Bellmann Ford algorithm, V times, once with each vertex as source with more sophisticated data structures.

- Complexity:
  - Around O(V * V * E)
  - In worst case can go upto $O(V^4)$ times
  -

All pair shortest path algorithm

Solution-3 : Floydd Warshalls algortihm

- Works with negative weight edges
- Assumes no negative weight cycles
- Complexity: $O(V^3)$

- A dynamic programming solution

All pair shortest path algorithm

Floydd Warshalls algortihm

Define a parameterized cost function that is easy to compute and also allows us to ultimately compute a final solution.

Cost function, $D^k_{i, j}$, which is defined as the distance from v_i to v_j using only intermediate vertices in the set $\{v_1, v_2, \ldots, v_k\}$

All pair shortest path algorithm

Floydd Warshalls algortihm

Initially, this Cost function, $D^{k}_{i,j}$, is given as

$$D^0_{i,j} = \begin{cases} 0 & \text{if } i = j \\ w((v_i, v_j)) & \text{if } (v_i, v_j) \text{ is an edge in } \vec{G} \\ +\infty & \text{otherwise.} \end{cases}$$

Given the initial value $D^{0}_{i,j}$, we can then easily define the value for an arbitrary k > 0 as

$$\begin{tikzcd} & k \arrow[dr, bend left] & \\ i \arrow[ur, bend left] \arrow[rr, "D"'] & & j \end{tikzcd}$$

$$D^k_{i,j} = \min\{D^{k-1}_{i,j}, \underbrace{D^{k-1}_{i,k} + D^{k-1}_{k,j}}\}.$$

All pair shortest path algorithm

Floydd Warshalls algortihm

Given the initial value $D^0_{i,j}$, we can then easily define the value for an arbitrary $k > 0$ as

$$D^k_{i,j} = \min\{D^{k-1}_{i,j}, D^{k-1}_{i,k} + D^{k-1}_{k,j}\}.$$

In simple terms,

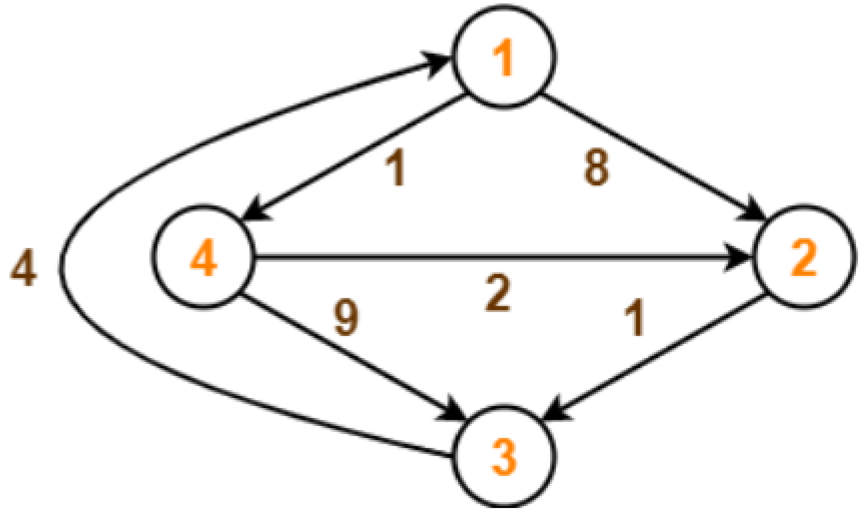
- One by one pick all vertices, say X
- Update all shortest paths which include the picked vertex, 'X' as an intermediate vertex in the shortest path.
- When we pick vertex number k as an intermediate vertex, we already have considered vertices {0, 1, 2, .. k-1} as intermediate vertices.
    - **A DP approach**

All pair shortest path algorithm

Input: A simple weighted directed graph $\vec{G}$ without negative-weight cycles
Output: A numbering $v_1, v_2, \ldots, v_n$ of the vertices of $\vec{G}$ and a matrix D, such that $D[i, j]$ is the distance from v_i to v_j in $\vec{G}$

let $v_1, v_2, \ldots, v_n$ be an arbitrary numbering of the vertices of $\vec{G}$
for $i \leftarrow 1$ **to** n **do**
  for $j \leftarrow 1$ **to** n **do**
    if $i = j$ **then**
      $D^0[i, i] \leftarrow 0$
    if (v_i, v_j) is an edge in $\vec{G}$ **then**
      $D^0[i, j] \leftarrow w((v_i, v_j))$
    else
      $D^0[i, j] \leftarrow +\infty$
for $k \leftarrow 1$ to n **do**
  for $i \leftarrow 1$ to n **do**
    for $j \leftarrow 1$ to n **do**
      $D^k[i, j] \leftarrow \min\{D^{k-1}[i, j], D^{k-1}[i, k] + D^{k-1}[k, j]\}$
return matrix D^n

All pair shortest path algorithm



$$D_0 = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \left[\begin{array}{cccc} 0 & 8 & \infty & 1 \\ \infty & 0 & 1 & \infty \\ 4 & \infty & 0 & \infty \\ \infty & 2 & 9 & 0 \end{array}\right] \end{array}$$

$i = j$ Put 0

No edge $= \infty$

otherwise $w[i, j]$

All pair shortest path algorithm

$$D_0 = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 8 & \infty & 1 \\ 2 & \infty & 0 & 1 & \infty \\ 3 & 4 & \infty & 0 & \infty \\ 4 & \infty & 2 & 9 & 0 \end{array}$$

$min = [d^{k-1}[i,j], \; d^{k-1}(i,k) + d^{k-1}[k,j]]$

$D_1 \Rightarrow$ keep vertex 1 as intermediate. keep R-1 & C-1 as it is

$$D_1 = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 8 & \infty & 1 \\ 2 & \infty & 0 & 1 & \infty \\ 3 & 4 & 12 & 0 & 5 \\ 4 & \infty & 2 & 9 & 0 \end{array}$$

$d^1(3,2) = min[d^0(3,2) \text{ or } d^0(3,1) + d^0(1,2)$
$= min[\infty \text{ or } (4+8)] = 12$

$d^1(3,4) = min[d^0(3,4) \text{ or } d^0(3,1) + d^0(1,4)]$
$= min[\infty \text{ or } (4+1)] = 5$

All pair shortest path algorithm

$$D_1 = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & & & & \\ 2 & & & & \\ 3 & & & & \\ 4 & & & & \end{array}\begin{bmatrix} 0 & 8 & \infty & 1 \\ \infty & 0 & 1 & \infty \\ 4 & 12 & 0 & 5 \\ \infty & 2 & 9 & 0 \end{bmatrix}$$

For D^2 keep ② as intermediate

Freeze R-2 & C-2

$d^2(1,3) = \min[d^1(1,3) \text{ or } d^1(1,2)+d^1(2,3)]$

$= \min[\infty, (8+1)] = 9$

$d^2(4,3) = \min[d^1(4,3) \text{ or } d^1(4,2)+d^1(2,3)]$

$= \min[9 \text{ or } (2+1)]$

$= 3$

$$D_2 = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & & & & \\ 2 & & & & \\ 3 & & & & \\ 4 & & & & \end{array}\begin{bmatrix} 0 & 8 & 9 & 1 \\ \infty & 0 & 1 & \infty \\ 4 & 12 & 0 & 5 \\ \infty & 2 & 3 & 0 \end{bmatrix}$$

All pair shortest path algorithm

$$D_2 = \begin{array}{cc} & \begin{matrix} 1 & 2 & 3 & 4 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \end{matrix} & \begin{bmatrix} 0 & 8 & 9 & 1 \\ \infty & 0 & 1 & \infty \\ 4 & 12 & 0 & 5 \\ \infty & 2 & 3 & 0 \end{bmatrix} \end{array}$$

$$D_3 = \begin{array}{cc} & \begin{matrix} 1 & 2 & 3 & 4 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \end{matrix} & \begin{bmatrix} & & 9 & \\ & & 1 & \\ 4 & 12 & 0 & 5 \\ & & 3 & \end{bmatrix} \end{array}$$

All pair shortest path algorithm

$$D_3 = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 8 & 9 & 1 \\ 2 & 5 & 0 & 1 & 6 \\ 3 & 4 & 12 & 0 & 5 \\ 4 & 7 & 2 & 3 & 0 \end{array}$$

$$D_4 = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & & & & 1 \\ 2 & & & & 6 \\ 3 & & & & 5 \\ 4 & 7 & 2 & 3 & 0 \end{array}$$