

4. FLOYD'S ALGORITHM

All-Pairs Shortest-Paths Problem:

Floyd's algorithm is an algorithm for finding shortest paths for all pairs in a weighted connected graph (undirected or directed) with (+/-) edge weights.

A **distance matrix** is a matrix (two-dimensional array) containing the distances, taken pairwise, between the vertices of graph.

The lengths of shortest paths in an $n \times n$ matrix D called the distance matrix: the element d_{ij} in the i th row and the j th column of this matrix indicates the length of the shortest path from the i th vertex to the j th vertex.

We can generate the distance matrix with an algorithm that is very similar to Warshall's algorithm is called Floyd's algorithm.

Floyd's algorithm computes the distance matrix of a weighted graph with n vertices through a series of $n \times n$ matrices:

$$D^{(0)}, \dots, D^{(k-1)}, D^{(k)}, \dots, D^{(n)}$$

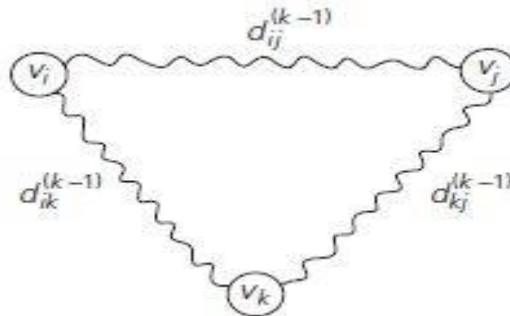
The element $d_{ij}^{(k)}$ in the i th row and the j th column of matrix $D^{(k)}$ ($i, j = 1, 2, \dots, n, k = 0, 1,$

\dots, n) is equal to the length of the shortest path among all paths from the i th vertex to the j th vertex with each intermediate vertex, if any, numbered not higher than k .

Steps to compute $D^{(0)}, \dots, D^{(k-1)}, D^{(k)}, \dots, D^{(n)}$

- The series starts with $D^{(0)}$, which does not allow any intermediate vertices in its paths; hence, $D^{(0)}$ is simply the weight matrix of the graph.
- As in Warshall's algorithm, we can compute all the elements of each matrix $D^{(k)}$ from its immediate predecessor $D^{(k-1)}$.
- The last matrix in the series, $D^{(n)}$, contains the lengths of the shortest paths among all paths that can use all n vertices as intermediate and hence is nothing other than the distance matrix.

Let $d_{ij}^{(k)}$ be the element in the i th row and the j th column of matrix $D^{(k)}$. This means that $d_{ij}^{(k)}$ is equal to the length of the shortest path among all paths from the i th vertex v_i to the j th vertex



v_j with their intermediate vertices numbered not higher than k .

FIGURE 3.4 Underlying idea of Floyd's algorithm.

The length of the shortest path can be computed by the following recurrence:

$$d_{ij}^{(k)} = \min\{d_{ij}^{(k-1)}, d_{ik}^{(k-1)} + d_{kj}^{(k-1)}\} \quad \text{for } k \geq 1, \quad d_{ij}^{(0)} = w_{ij}.$$

ALGORITHM Floyd($W[1..n, 1..n]$)

//Implements Floyd's algorithm for the all-pairs shortest-paths problem

//Input: The weight matrix W of a graph with no negative-length cycle

//Output: The distance matrix of the shortest

paths' lengths $D \leftarrow W$ //is not necessary if W can

be overwritten

for $k \leftarrow 1$ **to** n **do**

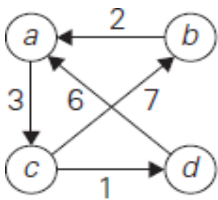
for $i \leftarrow 1$ **to** n **do**

for $j \leftarrow 1$ **to** n **do**

$D[i, j] \leftarrow \min\{D[i, j], D[i, k] + D[k, j]\}$

return D

Floyd's Algorithm's time efficiency is only $\Theta(n^3)$. Space efficiency is $\Theta(n^2)$. i.e. matrix size.



$$D^{(0)} = \begin{matrix} & \begin{matrix} a & b & c & d \end{matrix} \\ \begin{matrix} a \\ b \\ c \\ d \end{matrix} & \begin{bmatrix} 0 & \infty & 3 & \infty \\ 2 & 0 & \infty & \infty \\ \infty & 7 & 0 & 1 \\ 6 & \infty & \infty & 0 \end{bmatrix} \end{matrix}$$

$$D^{(1)} = \begin{matrix} & \begin{matrix} a & b & c & d \end{matrix} \\ \begin{matrix} a \\ b \\ c \\ d \end{matrix} & \begin{bmatrix} 0 & \infty & 3 & \infty \\ 2 & 0 & 5 & \infty \\ \infty & 7 & 0 & 1 \\ 6 & \infty & 9 & 0 \end{bmatrix} \end{matrix}$$

$$D^{(2)} = \begin{matrix} & \begin{matrix} a & b & c & d \end{matrix} \\ \begin{matrix} a \\ b \\ c \\ d \end{matrix} & \begin{bmatrix} 0 & \infty & 3 & \infty \\ 2 & 0 & 5 & \infty \\ 9 & 7 & 0 & 1 \\ 6 & \infty & 9 & 0 \end{bmatrix} \end{matrix}$$

$$D^{(3)} = \begin{matrix} & \begin{matrix} a & b & c & d \end{matrix} \\ \begin{matrix} a \\ b \\ c \\ d \end{matrix} & \begin{bmatrix} 0 & 10 & 3 & 4 \\ 2 & 0 & 5 & 6 \\ 9 & 7 & 0 & 1 \\ 6 & 16 & 9 & 0 \end{bmatrix} \end{matrix}$$

$$D^{(4)} = \begin{matrix} & \begin{matrix} a & b & c & d \end{matrix} \\ \begin{matrix} a \\ b \\ c \\ d \end{matrix} & \begin{bmatrix} 0 & 10 & 3 & 4 \\ 2 & 0 & 5 & 6 \\ 7 & 7 & 0 & 1 \\ 6 & 16 & 9 & 0 \end{bmatrix} \end{matrix}$$

Lengths of the shortest paths with no intermediate vertices ($D^{(0)}$ is simply the weight matrix).

Lengths of the shortest paths with intermediate vertices numbered not higher than 1, i.e., just **a** (note two new shortest paths from **b** to **c** and from **d** to **c**).

Lengths of the shortest paths with intermediate vertices numbered not higher than 2, i.e., **a** and **b** (note a new shortest path from **c** to **a**).

Lengths of the shortest paths with intermediate vertices numbered not higher than 3, i.e., **a**, **b**, and **c** (note four new shortest paths from **a** to **b**, from **a** to **d**, from **b** to **d**, and from **d** to **b**).

Lengths of the shortest paths with intermediate vertices numbered not higher than 4, i.e.,

a, **b**, **c**, and **d** (note a new shortest path from *c* to *a*).

FIGURE 3.5 Application of Floyd's algorithm to the digraph shown. Updated elements are shown in bold.