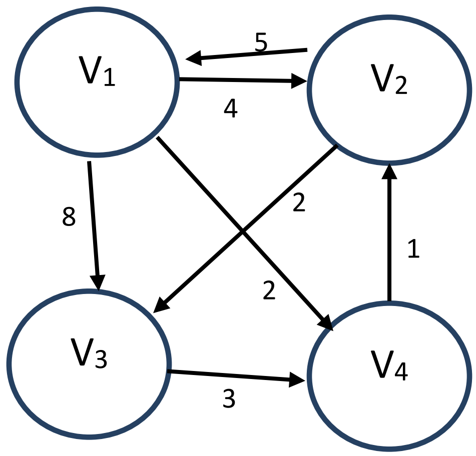
All Pairs Shortest Path Example

Finding Shortest Paths Between All Vertices in a Graph

Warshall-Floyd Algorithm



An adjacency matrix is used to represent the graph, and analyzed to find what is effectively a transitive closure.

The adjacency matrix is at right.

	V_1	V_2	V_3	V_4
V_1	0	3	5	2
V_2	5	0	2	5
V_3	9	4	0	3
V_4	6	1	3	0

The idea behind Warshall-Floyd is that there is a path from vertex i to vertex j if there is a path from i to k and a path from k to j.

We want to find minimum paths by cost. We will examine existing costs and compare with cost via vertex k.

	V_1	V_2	V_3	V_4
V_1	0	4	8	2
V_2	5	0	2	∞
V_3	∞	∞	0	3
V_4	∞	1	∞	0

- **Step 1:** Transfer all of the costs in the original matrix to the corresponding positions of a new adjacency matrix. The diagonal costs remain 0.

- **Step 2.** List separately the rows that have values in column k (=1 this time) and the columns that have 1's in row k.
  Rows: 2 Columns: 2, 3, 4

- **Step 3.** Pair each of the row numbers with each of the column numbers, and put the cost in the corresponding position of the matrix, if there isn't one with lower cost there already. Any remaining empty positions are left blank.

	V_1	V_2	V_3	V_4
V_1	0	4	8	2
V_2	5	0	2	7
V_3			0	3
V_4		1		0

(2,2), (2,3), (2,4) –

(2,2) has a 0; we can't do better. (2,3) has a 2. The alternative is v_2 to v_1 and v_1 to v_3. But, that costs 13, and the current cost is 2. We stick with the 2. But, for (2,4), there is no value (it was ∞); all the blanks are available to be filled in.

Using v_1 as the intermediary, we can go from v_2 to v_1 and v_1 to v_4. That cost is 5+2=7.

All Pairs Shortest Path Example

Finding Shortest Paths Between All Vertices in a Graph

Warshall-Floyd Algorithm

A graphic way of accomplishing this is by drawing boxes around row and column k.

We look at intersections, as depicted in the matrix, where the row and column both have a concrete value.

The 0 will remain. The 2 is less than the sum of the values in the row and column (5 and 8) and therefore remains. (v_2,v_4) is blank and therefore the sum will be placed, 5+2 = 7.

	V_1	V_2	V_3	V_4
V_1	0	4	8	2
V_2	5	0	**2**	7
V_3			0	3
V_4		1		0

We now work off the updated matrix.
Rows: 1,4   Columns: 1,3,4

(1,1) is 0; (1,3) is 8, but the two values that intersect are 4 and 2. 8 should become 6. Note that the 8 is at one of the intersections.
(1,4) has a 2, which is less than 11.

Now, Row 4. (v_4,v_2) has cost 1, and (v_2,v_1) has cost 5. Therefore, v_4 to v_1 has cost 6.

Using the graphic method, (v_4,v_3) has cost 3. Confirm this on the graph.

Complete the rest of this algorithm as an exercise.

	V_1	V_2	V_3	V_4
V_1	0	4	~~8~~ 6	2
V_2	5	0	**2**	7
V_3			0	3
V_4	6	1	3	0

K=3

	V_1	V_2	V_3	V_4
V_1	0	4	6	2
V_2	5	0	2	~~7~~ 5
V_3			0	3
V_4	6	1	3	0

K=4

	V_1	V_2	V_3	V_4
V_1	0	~~4~~ 3	~~6~~ 5	2
V_2	5	0	2	5
V_3	9	4	0	3
V_4	6	1	3	0

	V_1	V_2	V_3	V_4
V_1	0	3	5	2
V_2	5	0	2	5
V_3	9	4	0	3
V_4	6	1	3	0