Parallel Formulation and Computation:

Original Graph of 4 vertices

0	1	3	INF
INF	0	2	6
INF	8	0	1
INF	INF	INF	0

Partitioned into 2 subgraphs if 2 processors are used

0th and 1st column

The 0th column shows us all the edges from nodes 0,1,2,3 to 0th vertice respectively.
The 1st column shows us all the edges from nodes 0,1,2,3 to 1st vertice respectively.

0	1
INF	0
INF	8
INF	INF

2nd and 3rd column

The 0th column shows us all the edges from nodes 0,1,2,3 to 2nd vertice respectively.
The 1st column shows us all the edges from nodes 0,1,2,3 to 3rd vertice respectively.

3	INF
2	6
0	1
INF	0

Original k shortest distance matrix considering 0 as the source vertice and k = 3

0	INF	INF
INF	INF	INF
INF	INF	INF
INF	INF	INF

This will be partitioned into 2 subparts if we have 2 processors.

Processor 0 has the 0th and 1st column of the graph along with this distance matrix.

0	INF	INF
INF	INF	INF

Processor 1 has the 2nd and 3rd column of the graph along with this distance matrix.

INF	INF	INF
INF	INF	INF

Processor 0 will now compute the k shortest distance for the vertice 0 and vertice 1 using 0th and 1st column along **with communicating the priority queue with processor 1.**

The result will look like this for process 0 distance matrix

0	INF	INF
1	11	11

Processor 1 will now compute the k shortest distance for the vertice 2 and vertice 3 using the 2nd and 3rd column along **with communicating the priority queue with processor 0.**

The result will look like this for process 1 distance matrix

3	3	13
4	4	7

Both Process will now **gather** their distance matrix to give the final result as

0	INF	INF
1	11	11
3	3	13
4	4	7