KING ABDULAZIZ UNIVERSITY
FACULTY OF COMPUTING AND INFORMATION TECHNOLOGY
CPCS324 - Algorithms and Data Structures (II)
Spring 2021

Floyd-Warshall Algorithm

CPCS324 Project - Phase one

Instructor: Dr. Bassma Saleh Alsulami

Group 5

Name	Section
Dimah Abdullah Alolayan	DAR
Majd Saeed Gezan	DAR
Razan Muhammed Aljuhani	DAR

Table of Contents

Figures

1. Introduction

Algorithms in general play a very important role in the computer science field where it forms as its backbone, it is basically a step-by-step procedure to solve problems or to complete tasks. Before we dive into this report let us have a brief talk about one of the algorithm's types which is the Dynamic programming technique.

Dynamic Programming is one of the most powerful design techniques it is mainly used for solving optimization problems. especially, when the subproblems are not independent, but how does it work? It works as a Bottom-up approach where it solves all possible sub-problems and then combines them to achieve the best solutions for bigger problems so that it can be re-used whenever it is needed again.

This report focuses on one of the dynamic programming algorithms which is the Floyd-Warshall algorithm, the main idea behind this algorithm is finding the shortest path between all the pairs of vertices in a direct and indirect weighted graph, keeping in mind that it does not work for the graphs with negative cycles.

2. Screenshots for the outputs

```
run:
        CPCS324 – Project(Phase one) – Floyd–Warshall Algorithm

>>> The Given Weight Matrix:
D(0):
```

0	10	∞	∞	∞	5	∞	∞	∞	∞
∞	0	3	∞	3	∞	∞	∞	∞	∞
∞	∞	0	4	∞	∞	∞	5	∞	∞
∞	∞	∞	0	∞	∞	∞	∞	4	∞
∞	∞	4	∞	0	∞	2	∞	∞	∞
∞	3	∞	∞	∞	0	∞	∞	∞	2
∞	∞	∞	7	∞	∞	0	∞	∞	∞
∞	∞	∞	4	∞	∞	∞	0	3	∞
∞	∞	∞	∞	∞	∞	∞	∞	0	∞
∞	6	∞	∞	∞	∞	8	∞	∞	0

Figure 1: The Given Weight Matrix D(0)

On the k^{th} iteration, the algorithm determines shortest paths between every pair of vertices i, j that use only vertices among 1,..,k as intermediate, using the equation:

$$D^{k}[i,j] = \min\{D^{k-1}[i,j], D^{k-1}[i,k] + D^{k-1}[k,j]\}$$

- Comparing it with the index
- If the index is larger, it will change
  Otherwise, it remains the same.

The red highlight ( ) refers to the changed number.

Take the 1st row and 1st column from D(0) then use the equation to calculate D(1)

```
>>> The All Pairs Shortest path by Floyd's Algorithm:

D(1):
0   10  ∞   ∞   ∞   5   ∞   ∞   ∞   ∞
∞   0   3   ∞   3   ∞   ∞   ∞   ∞   ∞
∞   ∞   0   4   ∞   ∞   ∞   5   ∞   ∞
∞   ∞   ∞   0   ∞   ∞   ∞   ∞   4   ∞
∞   ∞   4   ∞   0   ∞   2   ∞   ∞   ∞
∞   3   ∞   ∞   ∞   0   ∞   ∞   ∞   2
∞   ∞   ∞   7   ∞   ∞   0   ∞   ∞   ∞
∞   ∞   ∞   4   ∞   ∞   ∞   0   3   ∞
∞   ∞   ∞   ∞   ∞   ∞   ∞   ∞   0   ∞
∞   6   ∞   ∞   ∞   ∞   8   ∞   ∞   0
```

Figure 2: D(1)

Take the 2nd row and 2nd column from D(1) then use the equation to calculate D(2)

```
D(2):
0   10  13  ∞   13  5   ∞   ∞   ∞   ∞
∞   0   3   ∞   3   ∞   ∞   ∞   ∞   ∞
∞   ∞   0   4   ∞   ∞   ∞   5   ∞   ∞
∞   ∞   ∞   0   ∞   ∞   ∞   ∞   4   ∞
∞   ∞   4   ∞   0   ∞   2   ∞   ∞   ∞
∞   3   6   ∞   6   0   ∞   ∞   ∞   2
∞   ∞   ∞   7   ∞   ∞   0   ∞   ∞   ∞
∞   ∞   ∞   4   ∞   ∞   ∞   0   3   ∞
∞   ∞   ∞   ∞   ∞   ∞   ∞   ∞   0   ∞
∞   6   9   ∞   9   ∞   8   ∞   ∞   0
```

Figure 3: D(2)

Take the 3rd row and 3rd column from D(2) then use the equation to calculate D(3)

```
D(3):
0   10  13  17  13  5   ∞   18  ∞   ∞
∞   0   3   7   3   ∞   ∞   8   ∞   ∞
∞   ∞   0   4   ∞   ∞   ∞   5   ∞   ∞
∞   ∞   ∞   0   ∞   ∞   ∞   ∞   4   ∞
∞   ∞   4   8   0   ∞   2   9   ∞   ∞
∞   3   6   10  6   0   ∞   11  ∞   2
∞   ∞   ∞   7   ∞   ∞   0   ∞   ∞   ∞
∞   ∞   ∞   4   ∞   ∞   ∞   0   3   ∞
∞   ∞   ∞   ∞   ∞   ∞   ∞   ∞   0   ∞
∞   6   9   13  9   ∞   8   14  ∞   0
```

Figure 4: D(3)

Take the 4th row and 4th column from D(3) then use the equation to calculate D(4)

D(4):

0	10	13	17	13	5	∞	18	21	∞
∞	0	3	7	3	∞	∞	8	11	∞
∞	∞	0	4	∞	∞	∞	5	8	∞
∞	∞	∞	0	∞	∞	∞	∞	4	∞
∞	∞	4	8	0	∞	2	9	12	∞
∞	3	6	10	6	0	∞	11	14	2
∞	∞	∞	7	∞	∞	0	∞	11	∞
∞	∞	∞	4	∞	∞	∞	0	3	∞
∞	∞	∞	∞	∞	∞	∞	∞	0	∞
∞	6	9	13	9	∞	8	14	17	0

Figure 5: D(4)

Take the 5th row and 5th column from D(4) then use the equation to calculate D(5)

D(5):

0	10	13	17	13	5	15	18	21	∞
∞	0	3	7	3	∞	5	8	11	∞
∞	∞	0	4	∞	∞	∞	5	8	∞
∞	∞	∞	0	∞	∞	∞	∞	4	∞
∞	∞	4	8	0	∞	2	9	12	∞
∞	3	6	10	6	0	8	11	14	2
∞	∞	∞	7	∞	∞	0	∞	11	∞
∞	∞	∞	4	∞	∞	∞	0	3	∞
∞	∞	∞	∞	∞	∞	∞	∞	0	∞
∞	6	9	13	9	∞	8	14	17	0

Figure 6: D(5)

Take the 6th row and 6th column from D(5) then use the equation to calculate D(6)

D(6):

0	8	11	15	11	5	13	16	19	7
∞	0	3	7	3	∞	5	8	11	∞
∞	∞	0	4	∞	∞	∞	5	8	∞
∞	∞	∞	0	∞	∞	∞	∞	4	∞
∞	∞	4	8	0	∞	2	9	12	∞
∞	3	6	10	6	0	8	11	14	2
∞	∞	∞	7	∞	∞	0	∞	11	∞
∞	∞	∞	4	∞	∞	∞	0	3	∞
∞	∞	∞	∞	∞	∞	∞	∞	0	∞
∞	6	9	13	9	∞	8	14	17	0

Figure 7: D(6)

Take the 7th row and 7th column from D(6) then use the equation to calculate D(7)

D(7):

0	8	11	15	11	5	13	16	19	7
∞	0	3	7	3	∞	5	8	11	∞
∞	∞	0	4	∞	∞	∞	5	8	∞
∞	∞	∞	0	∞	∞	∞	∞	4	∞
∞	∞	4	8	0	∞	2	9	12	∞
∞	3	6	10	6	0	8	11	14	2
∞	∞	∞	7	∞	∞	0	∞	11	∞
∞	∞	∞	4	∞	∞	∞	0	3	∞
∞	∞	∞	∞	∞	∞	∞	∞	0	∞
∞	6	9	13	9	∞	8	14	17	0

Figure 8: D(7)

Take the 8th row and 8th column from D(7) then use the equation to calculate D(8)

D(8):

0	8	11	15	11	5	13	16	19	7
∞	0	3	7	3	∞	5	8	11	∞
∞	∞	0	4	∞	∞	∞	5	8	∞
∞	∞	∞	0	∞	∞	∞	∞	4	∞
∞	∞	4	8	0	∞	2	9	12	∞
∞	3	6	10	6	0	8	11	14	2
∞	∞	∞	7	∞	∞	0	∞	11	∞
∞	∞	∞	4	∞	∞	∞	0	3	∞
∞	∞	∞	∞	∞	∞	∞	∞	0	∞
∞	6	9	13	9	∞	8	14	17	0

Figure 9: D(8)

Take the 9th row and 9th column from D(8) then use the equation to calculate D(9)

D(9):

0	8	11	15	11	5	13	16	19	7
∞	0	3	7	3	∞	5	8	11	∞
∞	∞	0	4	∞	∞	∞	5	8	∞
∞	∞	∞	0	∞	∞	∞	∞	4	∞
∞	∞	4	8	0	∞	2	9	12	∞
∞	3	6	10	6	0	8	11	14	2
∞	∞	∞	7	∞	∞	0	∞	11	∞
∞	∞	∞	4	∞	∞	∞	0	3	∞
∞	∞	∞	∞	∞	∞	∞	∞	0	∞
∞	6	9	13	9	∞	8	14	17	0

Figure 10: D(9)

Take the 10th row and 10th column from D(9) then use the equation to calculate D(10) and it is the final result

```
D(10):
0       8       11      15      11      5       13      16      19      7
∞       0       3       7       3       ∞       5       8       11      ∞
∞       ∞       0       4       ∞       ∞       ∞       5       8       ∞
∞       ∞       ∞       0       ∞       ∞       ∞       ∞       4       ∞
∞       ∞       4       8       0       ∞       2       9       12      ∞
∞       3       6       10      6       0       8       11      14      2
∞       ∞       ∞       7       ∞       ∞       0       ∞       11      ∞
∞       ∞       ∞       4       ∞       ∞       ∞       0       3       ∞
∞       ∞       ∞       ∞       ∞       ∞       ∞       ∞       0       ∞
∞       6       9       13      9       ∞       8       14      17      0

BUILD SUCCESSFUL (total time: 0 seconds)
```

Figure 11: D(10) Final Result

3. Difficulties faced during the phase design

One of the difficulties that faced us during this project is coding a new concept and implementing it, we overcame it by researching and investigating more on it. We also had difficulty in tracing the matrix due to its large size which made it a little bit complex to trace, lastly the difficulty in meeting in-person due to COVID-19, we overcame it by staying in contact regularly with online meetings, we used screen sharing and annotation to enable everyone to stay focused and learn everything we needed to.

4. Conclusion

To conclude, in this report we used Floyd-Warshall algorithm that uses dynamic programming to get the optimized solution. However, we had a brief discussion about dynamic programming and the main purpose of Floyd-Warshall algorithm which is finding the shortest path between all the pairs of vertices in a direct and indirect weighted graph.

5. References

Levitin, A. (2012,2007,2003). *Introduction to The Design & Analysis of Algorithms.* New Jersey: Addison-Wesley.

https://www.javatpoint.com/dynamic-programming-introduction