Jagannath University Journal of Science
Volume 11, Number I, Jun. 2024, pp. 45−52
https://jnu.ac.bd/journal/portal/archives/science.jsp
ISSN 3005-4486 (Online), ISSN 2224-1698 (Print)

Jagannath University Journal of Science

Improved Floyd-Warshall Algorithm for Solving Travelling Salesman Problem

Research Article

Solima Khanam*, Shourov Roy and Musfiqur Rahman Shihab
Department of Mathematics, Jagannath University, Dhaka-1100, Bangladesh

DOI: https://doi.org/10.3329/jnujsci.v11i1.76692

Received: 14 January 2024, Accepted: 26 May 2024

ABSTRACT

The shortest path problem is a fundamental challenge in graph theory, focused on identifying the most efficient routes between nodes in a network. It stands as one of the extensively researched combinatorial optimization problems. In this paper, we explained the fundamental concepts of shortest path algorithms, with a particular emphasis on Floyd Warshall algorithm. We apply the algorithm and showed how this algorithm can be effectively employed to determine the shortest path in various scenarios. Furthermore, this paper addresses real-life applications, particularly focusing on the traveling salesman problem. Individuals often have the option to travel along different routes to reach their destination. However, selecting suboptimal routes can result in time-consuming journeys. To tackle this issue, we apply the Floyd Warshall algorithm to solve the traveling salesman problem (TSP). Additionally, we demonstrate the enhancement of the Floyd Warshall algorithm's efficiency for TSP using the rectangular algorithm. This paper concludes with a comparative analysis between the original and improved Floyd Warshall algorithms, emphasizing the computational reduction achieved through the proposed enhancements.

Keywords: *Shortest path, Travelling salesman problem (TSP), Floyd Warshall algorithm, NP hard problem*

1. Introduction

The shortest path problem is a classic optimization challenge that revolves around determining the most efficient route between two nodes in a network (Llanos et al. 2014; Hart et al. 1968). The primary objective is to identify the path with the minimal cost, with the cost being quantifiable in terms of distance, time, or any other relevant metric. Solving the shortest path problem involves employing algorithms to identify the optimal edge

Corresponding author: Solima Khanam
E-mail: solimakhanam@yahoo.com

path between vertices in a graph. Several algorithmic variations exist for determining the pursued node based on the given graph's direction (Roy 1959). These variations include single-pair, single-source, single-destination, and all-pairs shortest path problems. In the single-pair shortest path problem, the objective is to compute the shortest path between a specified pair of vertices. Noteworthy algorithms for addressing this problem include Dijkstra's and the Bellman-Ford algorithm (Ford et al.1962; Ford 1956), both renowned for solving single-source shortest path problems. Conversely, the all-pairs shortest path problem entails computing the shortest path from every pair of vertices in the graph. For this category, notable algorithms include the Floyd-Warshall algorithm and Johnson's algorithms (Floyd et al.1962; Warshall 1962; Khamami et al. 2019; Habib et al. 2023), recognized for their efficacy in solving all-pairs shortest path problems.

One of the most renowned examples of a shortest path algorithm is the Traveling Salesman Problem (TSP). The primary objective of the TSP is to minimize the overall distance traveled, with the constraint that each city must be visited precisely once (Amin et al. 1976).

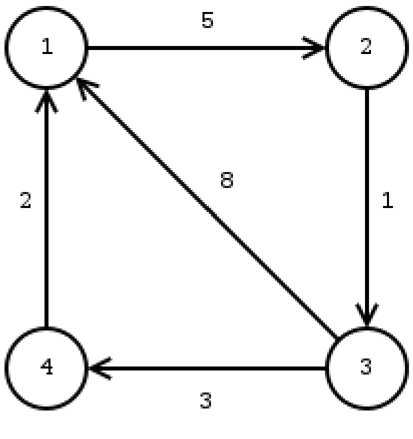

Fig.1 Travelling Salesman Problem

In computer science, the Traveling Salesman Problem (TSP) stands as a well-known and significant combinatorial optimization challenge. The objective of TSP is to identify the shortest tour that visits each city in a specified list exactly once and subsequently returns to the starting city. The primary focus of TSP is to determine the path with the least cost, measured in terms of distance. Originating in 1930, this problem has been the subject of extensive research and is regarded as one of the most intensely studied optimization problems (Amin et al. 1976; Barachet, 1977; Rosen 2003).

The TSP is effectively modeled as a weighted graph, wherein the vertices correspond to cities, and the weights on the edges denote the travel costs between pairs of cities. Solving the TSP involves identifying a path that traverses all graph vertices exactly once, with the minimal total weight.

The primary drawback of the Traveling Salesman Problem lies in its classification as an NP-hard problem. This categorization implies that finding an optimal solution becomes exponentially time-consuming as the size of the problem increases. Consequently, this exponential growth in computational demand renders the problem impractical for solving large instances, thereby limiting its real-world applicability. Moreover, the problem's assumption of constant travel costs between locations may not always align with real-world scenarios where travel costs can vary. Additionally, the Traveling Salesman Problem is based on a closed universe premise, mandating that all locations must be visited exactly once. This assumption may not always be suitable for certain applications (Aini et al. 2012). However, Dijkstra's algorithm, following a Greedy Approach (Lestari 2017), encounters limitations. Once a node is marked as visited, it cannot be reconsidered, even if an alternative path with a lesser cost or distance exists. This becomes problematic in the presence of negative weights or edges in the graph. Consequently, Dijkstra's algorithm may fail to find the minimum distance. In such cases, the Bellman-Ford algorithm is a suitable alternative, especially when dealing with negative weights, as it halts the loop upon encountering a negative cycle. To address the challenges posed by negative weights, we have adopted the Floyd-Warshall algorithm. To enhance the time complexity of Floyd-Warshall, the rectangular algorithm is applied, contributing to the algorithm's improved performance.

In this paper, we apply Floyd-Warshall algorithm combined with rectangular algorithm ((Aini et al. 2012) to enhance the performance of Floyd-Warshall algorithm to make the computation of Travelling salesman problem (TSP) easier. Aini et al. applied rectangular algorithm for cyclic graph with a better performance. In the TSP (travelling salesman problem) problem, we usually assume a complete graph. Here, we have shown an example of a travelling salesman problem of not complete graph. The improved Floyd-Warshall algorithm produces an excellent performance in case of TSP. Moreover, experiment shows that the number of calculations remain unchanged with the increment of nodes comparing to general Floyd-Warshall algorithm. Ultimately, the improved Floyd-Warshall algorithm demonstrated superior performance in calculating the shortest path for the traveling salesman problem for with, achieving reduced computational demands.

The rest of the paper is organized as follows: in Section 2, Floyd Warshall algorithm and rectangular algorithm is discussed and then experiment is conducted using improved Floyd Warshall for travelling salesman problem in Section 3. A comparative performance analysis is shown (calculation vs. number of nodes) travelling salesman problem with 7 nodes. Finally, we draw a conclusion in Section 4.

2. Method

In this section we have Floyd Warshall algorithm and Rectangular algorithm is discussed and then Floyd Warshall is improved using rectangular algorithm.

2.1 Floyd Warshall algorithm

The Floyd-Warshall algorithm, also called Floyd's algorithm, is a computer science technique that effectively and simultaneously finds the shortest paths between each pair of vertices in a weighted and possibly directed graph. As long as there are no negative cycles, this graph can have either positive or negative edge weights. One prominent and frequently used example of a dynamic programming approach that is well-known for its simplicity and ease of implementation is the Floyd-Warshall algorithm (Cormenet et al.1990; Habib et al. 2023).

Algorithm:

Step 1: Initialize two square matrices D_j and R_j, where j represents the stage number and n is the total number of nodes in the network.

Step 2: For j = 0, calculate initial matrices D_0 and R_0 as follows:

- D_0 is a matrix where each element d_{ik} is set to:
  - $d_{ik} = d_{ik}$ if there is a direct route connecting node i to node k
  - ∞ if there is no direct route connecting node i to node k
  - 0 if i = k
- R0 is a matrix where each element r_{ik} is set to:
  - k if there is a direct route connecting node i to node k
  - '-' if there is no direct route connecting node i to node k
  - '-' if i = k

Step 3: For the remaining stages j = 1 to n, calculate matrices D_j and R_j as follows:

- D_j is a matrix where each element d_{ik} is calculated as:

  - $d_{ik} = d_{ik}$ if $i = k$ or $i = j$ or $k = j$
  - $\min(d_{ik}, d_{ij} + d_{jk})$ otherwise
- R_j is a matrix where each element r_{ik} is calculated as:
  - k if $i = k$ or $i = j$ or $k = j$
  - k if $d_{ik} \leq d_{ij} + d_{jk}$
  - j if $d_{ik} > d_{ij} + d_{jk}$

Step 4: Repeat Step 3 until matrices D_n and R_n are obtained.

In this section, we present the novelty our work. Will apply Floyd-Warshall algorithm combined with rectangular algorithm to enhance the performance of Floyd-Warshall algorithm to make the computation of travelling salesman problem (TSP) easier. The Rectangular algorithm is described below.

2.3 Rectangular Algorithm

The algorithm is named after the rectangular graphical approach it employs, not only does it demand less computational resources and boast straightforward implementation, but its simplicity makes it particularly suitable for educational contexts (Aini et al. 2012). By reducing the computational load of the Floyd-Warshall algorithm, this approach enhances efficiency, resulting in a more streamlined and easier-to-implement version of the algorithm.

Algorithm

Algorithm:

Step 1: Initialize D_j and R_j as N×N square matrices, where j represents the stage number, N is the total number of nodes, D is the distance matrix, and R is derived from D.

Step 2: Compute the initial matrices D_0 and R_0 using the Floyd–Warshall algorithm.

Step 3: Calculate the D_j matrices for j = 1 to N:

- Firstly, retain unchanged entities adjacent to an '8' in a row or column of the D_j matrix by replacing them with values from D_{j-1}, providing a significant speedup. If a complete D_j matrix is not achieved, derive the remaining entities using a set of rectangles.
- Secondly, accelerate the procedure by deriving the R_j matrix as follows:
  - If an entity in D_j remains unchanged, the corresponding entity in R_j will not change.
  - If an entity changes in D_j, substitute its associated entity in R_j with j.

Step 4: Repeat Step 3 until matrices D_n and R_n are obtained.

3.1 Experiments with Travelling Salesman Problem (TSP)

In this section we solved a travelling salesman problem with 7 nodes and explain

How the speed up process works in every

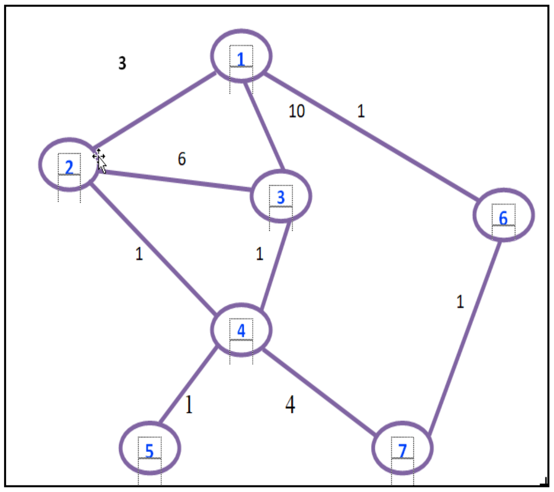

Fig. 2 Solution to shortest path (7 nodes) problem

The operation of the Floyd-Warshall algorithm has already been demonstrated. We will now use a path with seven nodes (fig. 2) to test the method. As we proceed with the example, we shall explain the rectangular algorithm. If n equals 7, we require two square 7 × 7 matrices, D_j and R_j, for eight stages, ranging from stage '0' to stage '7'. Step 2 of the rectangular algorithm can be used to derive the matrices D_0 and R_0:

D_0=

Nodes	1	2	3	4	5	6	7
1	0	3	10	∞	∞	1	∞
2	3	0	6	1	∞	∞	∞
3	10	6	0	1	∞	∞	∞
4	∞	1	1	0	1	∞	4
5	∞	∞	∞	1	0	∞	∞
6	1	∞	∞	∞	∞	0	1
7	∞	∞	∞	4	∞	1	0

R_0=

Nodes	1	2	3	4	5	6	7
1	-	2	3	-	-	6	-
2	1	-	3	4	-	-	-
3	1	2	-	4	-	-	-
4	-	2	3	-	5	-	7
5	-	-	-	4	-	-	-
6	1	-	-	-	-	-	7
7	-	-	-	4	-	6	-

When j = 1, the first row and first column of the D1 matrix are exactly the same as the first row and first column of the D_0 matrix when applying the rectangular method in order to produce the D_1 and R_1 matrices (keep in mind that the diagonal values stay the same throughout all stages). Thus we have to recalculate only d_{23}, d_{24}, d_{25},d_{26},d_{27},d_{32},d_{34},d_{35},d_{36},d_{37},d_{42},d_{43},d_{45},d_{46},d_{47}, d_{52},d_{53},d_{54},d_{56},d_{57},d_{62},d_{63},d_{64},d_{65},d_{67},d_{72},d_{73},d_{74}, d_{75} and d_6. After Speed up Procedure 1: the entities of the seventh row and seventh column of the D_0 matrix will appear intact (Fig. 2) in the D_1 matrix since there is ∞ in both the first row and the first column of the D_0 matrix. Therefore, only d_{23}, d_{32}, d_{26}, d_{62}, d_{36}, and d_{63} need to be adjusted. Since we are in stage 1 (j = 1), we build a rectangle in D_0 beginning at d_{11} in order to calculate d_{23}. D_{23} would be the corner on the other side. With these two corners, we may build the rectangle as it is depicted in. $D_{23} = \min(d_{23}, d_{21}+d_{13}) = \min(6,3+10) = 6$ in D_1, as a result. In a similar manner, $d_{32} = \min(6, 10 + 3)$, $d_{26} = \min(\infty,3+1)$, $d_{62} = \min(\infty,1+3)$, $d_{36} = (\infty,10+1)$, and $d_{63} = (\infty,1+10)$.

Thus D_1=

Nodes	1	2	3	4	5	6	7
1	0	3	10	∞	∞	1	∞
2	3	0	6	1	∞	4	∞
3	10	6	0	1	∞	11	∞
4	∞	1	1	0	1	∞	4
5	∞	∞	∞	1	0	∞	∞
6	1	4	11	∞	∞	0	1
7	∞	∞	∞	4	∞	1	0

R_1=

Nodes	1	2	3	4	5	6	7
1	-	2	3	-	-	6	-
2	1	-	3	4	5	1	7
3	1	2	-	4	5	1	7
4	-	2	3	-	5	6	7
5	-	2	3	4	-	6	7
6	1	2	3	4	5	-	7
7	-	2	3	4	5	6	-

However, a complete calculation for deriving only matrices D_1 and R_1 is as follows. Applying step 3 of the Floyd–Warshall algorithm to derive D_1 would result in the following values. Note that the first row and the first column (j = 1) would be the same. Furthermore, the diagonal also remains the same.

d_{23}=min (d_{23}, d_{21}+d_{13}) = min (6,3+10) = 6

d_{32} =min(d_{32},d_{31}+d_{12})= min(6, 10 + 3) = 6

d_{26}=min(d_{26},d_{21}+d_{16})=min(∞,3+1) = 4

d_{62}=min(d_{62},d_{61}+d_{12})=min(∞,1+3) = 4

d_{36}=min(d_{36},d_{31}+d_{16})=min(∞,10+1) = 11

and d_{63}=min (d_{63}, d_{61}+d_{13}) = min (∞,1+10) = 11

following values: r_{12} = k = 2 ,r_{13} = k = 3 ,r_{16} = k = 6.

r_{21} = k = 1 , r_{23} = k= 3, r_{24} = k =4, r_{25}=k =5, r_{26}=j=1,r_{27}=k=7.

r_{31}=k=1,r_{32} = k =2,

r_{34}=k=4, r_{35}=k=5,r_{36}=j=1,r_{37}=k=7.

r_{42}=k=2,r_{43}=k=3,r_{45}=k=5,r_{46}=k=6,r_{47}=k=7.

r_{52}=k=2, r_{53}=k=3, r_{54}=k=4, r_{56}=k=6,r_{57}=k=7.

r_{61}=k=1,r_{62}=k=2,r_{63}=k=3,r_{64}=k=4,r_{65}=k=5,r_{67}=k=7

r_{72}=k=2,r_{73}=k=3,r_{74}=k=4,r_{75}=k=5,r_{76}=k=6.

similarly, we have the (D_2:R_2), (D_3:R_3), (D_4:R_4), (D_5:R_5), (D_6:R_6), (D_7:R_7) matrices and the last iterative unchanged. We have use C programming codes to get matrices D_j and R_j. After computing (D_7:R_7) matrix we found the shortest or minimum distance that salesman must travel is 23 units; one of the best path is :1→3→2→4→7→5→6→1.

D_7=

Nodes	1	2	3	4	5	6	7
1	0	3	5	4	5	1	1
2	3	0	2	1	2	4	5
3	5	2	0	1	2	6	5
4	4	1	1	0	1	5	4
5	5	2	2	1	0	6	5
6	1	4	6	5	6	0	1
7	1	5	5	4	5	1	0

R_7=

Nodes	1	2	3	4	5	6	7
1	-	2	3	4	5	6	7
2	1	-	3	4	5	6	7
3	1	2	-	4	5	6	7
4	1	2	3	-	5	6	7
5	1	2	3	4	-	6	7
6	1	2	3	4	5	-	7
7	1	2	3	4	5	6	-

3.2 Graph showing number of calculations vs. number of nodes

In this work, we introduced a novel approach for calculating the shortest path in networks with cycles. A simulation study was conducted to evaluate the computational performance of two algorithms, namely the Floyd-Warshall algorithm and the rectangular algorithm. From the experiment we observed that if the number of the nodes increase in a travelling salesman problem then, calculation will increase in case of Floyd-warhall algorithm. However, increase of improved Floyd-Warshall algorithm. calculation remain unchanged with the increment of nodes in a problem (Fig.3).

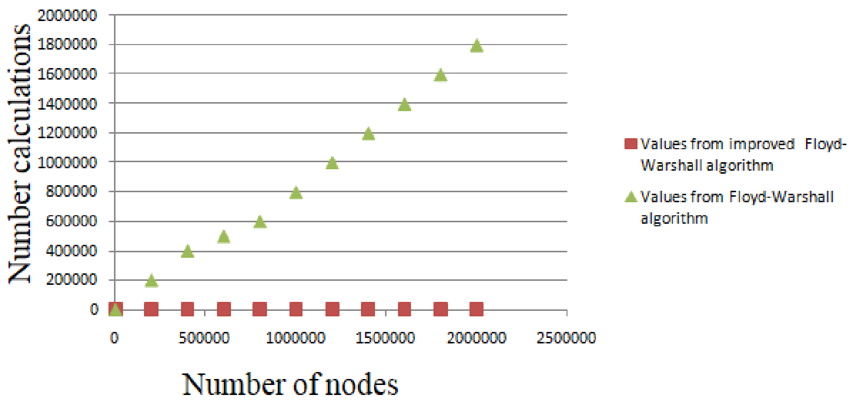

Fig.3. Number of calculations vs. number of nodes

Actually, the proposed rectangular algorithm, improves the Floyd–Warshall algorithm, in a number of ways. The Floyd–Warshall algorithm and the rectangular algorithm have exactly the same performance in deriving the D_0 and the R_0 matrices. For the stages $j \geq 1$, however, the rectangular algorithm derives the associated matrices much more quickly due to the reduced amount of calculation.

4. Conclusion

This paper explores the fundamental concepts of Floyd Warshall algorithms. It demonstrates how the algorithm can be applied to determine the shortest path in a real-life problem, particularly the Traveling Salesman Problems are discussed with the application of the Floyd Warshall algorithm for its solution. The study includes experiments using the Floyd-Warshall algorithm, acknowledging its NP-hard nature. To enhance its performance, the rectangular algorithm is employed. The improved Floyd-Warshall algorithm demonstrates superior performance in calculating the shortest path for the Traveling Salesman Problem (TSP), achieving reduced computational requirements. As a direction for future research, further investigation into enhancing the rectangular algorithm is underway, aiming to minimize the calculation needed when deriving the D_j and the R_j matrices.

References

Aini A, Salehipour A. 2012. Speeding up the Floyd–Warshall algorithm for the cycled shortest path problem. *Applied Mathematics Letters*, 25(1).

Amin AR, Ikshan, M. and Wibisono, L. 1976. Travelling Salesman Problem, *Journal of the Physics*: conf:1-6.

Barachet LL. 1957. Letter to the editor- Graphic Solution of Travelling Salesman Problem, *Operation Research*. 5(6):841-845.

Bellman RE. 1958. On a routing problem, *Quarterly of Applied Mathematics.*16 (1): 87–90.

Cormen TH, Leiserson CE, Rivest RL and Stein C. 2001. Introduction to Algorithms (2nded). *MIT Press and McGraw Hill*: 595-601.

Cormen TH, Leiserson CE. and Rivest RL. 1990. Introduction to Algorithms (1sted), *MIT Press and McGraw Hill*, 26(2):558-565.

Floyd RW. 1962. Algorithm 97: *Shortest Path, Communication of the ACM*, 5(6):345.

Ford LR. 1956. Network flow theory. *Santa Monica*: 923.

Ford LR, Fulkerson DR. 1962. Flows in Networks. *Princeton University Press*, *Princeton*.

Habib S, Muhammad, M. A. and Mohammed, S. 2023. Floyd-Warshall Algorithm Based on Picture Fuzzy Information, *Computer Modeling in Engineering & Sciences*.

Hart PE, Nilsson NJ. and Raphael B. 1968. A Formal Basis for the heuristic Determination of Minimum costs Paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100-107.

Khamami A, Saputra, R. 2019. The shortest path search application based on the city transport route in Semarang using the Floyd- warshall algorithm. *Journal of Physics*: *Conference Series*, 1217(1).

Lestari S, Puspa AK. 2017. Analysis Determination of Shortest Route Delivery Using Dijkstra Algorithm. *The 4th International Conference on Engineering and Technology Development* (ICETD 2017) :308-3018.

Llanos DR, Escribano AG, and Arranz HO. 2014. The Shortest Path Problem: Analysis and Comparison of Methods. Synthesis *Lectures on Theoretical Computer Science*, 1(1):1-87.

Roy B. 1959. Transitivity and Connectedness, *C. R. Acad. Sci. Paris*: 216-218.

Rosen KH. 2003. Discrete Mathematics and its applications. *McGraw Hill*, 9(69):657.

Warshall S. 1962. A Theorem on Boolean Matrices, *Journal of the ACM*. 9:11–12.