Optimum Route Finding In Tourism Transportation in Tamil Nadu Using Weighted Fuzzy Graph

Santhoshkumar.S[1] , P.Nedumaran[2] , M.Peter[3] , J.Jose Stephy[4] , J.Janet Sheeba[5]
[1]PG Department of Mathematics, Patrician College of Arts and Science, Chennai
[2]PG and Research Department of Mathematics, Guru Nanak College, Chennai
[3,4,5]PG and Research Department of Mathematics, Loyola College, Chennai
Email: santhoshkumar@patriciancollege.ac.in[1], periyanedumaran@gmail.com[2], matzpeter@loyolacollege.edu[3], jsjosestephy@gmail.com[4], jjanetsheeba@loyolacollege.edu[5]

Received: 11.07.2024 Revised: 22.08.2024 Accepted: 27.09.2024

ABSTRACT
Traveling opens a significant door for us to delve into the world beyond our imagination and engage in various activities, giving rise to the concept of tourism. Tourism has evolved as a major economic activity over the years. To explore tourist destinations, it is crucial to identify the most efficient way to minimize both time and expenses. Utilizing data on tourist arrivals in Tamil Nadu, a graph G is constructed to represent the major tourist spots, and the edges' weights are determined using fuzzy membership functions. The pre-existing graph G is then subjected to the Floyd-Warshall Algorithm to ascertain the optimal route. This study aims to identify the optimal route between tourist spots in Tamil Nadu, considering factors such as distance, time, and travel expenses. The results provide insights into exploring Tamil Nadu in the most efficient manner possible.

Keywords: Floyd-Warshall, world, Tourism, optimal.

1. INTRODUCTION

Tamil Nadu boasts a well-developed, dense, and modern transportation infrastructure encompassing public and private transport. With an extensive road network spanning approximately 2.71 lakh km, connecting cities, towns, and major villages, transportation growth has significantly contributed to the development of tourism in the state. The comprehensive road network in Tamil Nadu, comprising both surfaced and unsurfaced roads, spans an impressive length of nearly 1.82 lakh km. This well- connected road system enhances accessibility and connectivity across the state.Tourism not only sustains age-old cultures and traditions but also offers local residents additional opportunities and avenues to earn a livelihood. It adds value by showcasing their cultural heritage, arts, crafts, and involves performances such as dance, drama, and other artistic expressions.

Moreover, the potential for exploring inland cruise services on rivers and canals is recommended for further exploration, adding yet another dimension to the state's transportation offerings. Despite this progress, there is a need for personalized travel route planning from the perspective of tourists, rather than relying solely on tourism intermediaries. This study aims to address this gap by listing the shortest routes to tourist spots in Tamil Nadu, facilitating easy customization of travel itineraries.

This algorithm is to find the shortest path in directed weighted graph with either positive or negative edge weights. This algorithm assists to find the length of the shortest paths between all pairs of vertices. Whereas it does not return details of the paths, it is very practicable to reconstruct the paths with easy modifications to the algorithm logic. "In general, this algorithm is a deviation of dynamic programming, a technique which performs the troubleshooting, by viewing the solutions that would be obtained as inter-related decisions. In order to find the shortest path iterations,this algorithm begin from the starting point and then expand the path by evaluating each and every point one by one till it reached the destination point with the amount of weight to a minimum.

The input of this algorithm is a matrix graph and its output circuit is the shortest path from all points."Floyd-Warshall algorithm compares all possible trajectories on the graph for each line of all points. Assume that there exists a graph G with vertices V, each numbered from 1 to n. Assume that there is also a function shortest Path (i, j, k), which restores the possibility of a shortest path from i to j by utilize only the node 1 to k as a conciliator point.

The ultimate purpose of use of this function is to find the shortest path from each vertex i to vertex j by an intermediary node 1 to k + 1.

There are two possible options are
1. "The shortest path is in fact currently come from the nodes that are between 1 and k."
2. "There are a few line from the vertices i to k + 1, and also from k + 1 to j."
"Point that the shortest path from i to j which is now precedent the nodes 1 to k defined in function shortest Path (i, j, k) and it is clear that if there is a solution from i to k + 1 to j, then the length of the solution had been a the number of shortest paths from i to k + 1 and the shortest path from k + 1 to j. Hence, the algorithm for the function shortest Path (i, j, k) are as follows:"

Algorithm: "Fuzzy Floyd-Warshall "
"Entries (i, j) of the matrix provides the weights wIJ from nodes i to j, which is finite if i is linked directly to j, and infinite otherwise. The design of the Floyd's algorithm is simple and clear-cut. Given three nodes i, j and k with the connecting weights on three edges, it is shorter to reach j from I passing through k if wik+ wkj<wij . In this scenario, it is optimal to reserve the direct route from i → j with the not direct route i→k→j.

This operation is applied logically to the network by using following steps
Initial Stage: Define the starting weight matrix W0 and the node sequence matrix S0 as mentioned below. The diagonal elements are marked with (-) to indicate that they are blocked. Set k = 1.
General Step k: Define row k and column k as pivot row and pivot column respectively. Apply the triple operation to each element wij in wk-1 , for all i and j.
If the condition is wik+ wkj<wij for different i, j and k is fulfilled then implement following changes in Wk and Sk .
Create Wk by replacing wij in Wk-1 with wik+ wkj<wij. Create Sk by replacing sij in Sk-1 with k. Set k = k +1. If k=n then stop, else repeat step k."
After n steps, we can determine the shortest route between nodes i and j from the matrices Wn and Sn using following rules:
1. From Wn ,wij gives the weights of shortest route between node i and j.
2. From Sn, determine the intermediate node k = sij that yield the route i →k →j.
If sik = k and skj =j , stop; all intermediate nodes of shortest route has been found, if not, repeat the procedure between nodes i and k, and between k and j.

2. LITERATURE REVIEW

The shortest path problem, a fundamental challenge with widespread applications, finds its solution through the application of Graph theory. In the year of 2014, the scholars H. Pandey and P. P. Pande [1] have underscored the significance of graph theory through the encounter of real-world issues. They emphasize its relevance in operations research and computer science, showcasing existing applications. A prominent and widely utilized algorithm for computing the shortest path from an origin to a destination is Dijkstra's algorithm.
The challenge of calculating edges with uncertain weights renders Dijkstra's algorithm inefficient. Consequently, the importance of the Floyd-Warshall Algorithm comes to the forefront. This method excels in handling problems where the solution is intricately interconnected. Researcher Edsger W. Dijkstra, [2] has discued Problems in Connexion with Graphs, the single source, single destination shortest path problem paved the way for Dijkstra's algorithm in 1959. However, Floyd and Warshall's algorithm emerged as a more versatile solution for scenarios involving uncertain edgeweights. I n 1 9 6 2 , t he algorithm was formally presented in an article titled Algorithm 97: Shortest Path. [3]
Researchers, Zuhry, Andysah, and Mesran [4] explored the topic of identifying the shortest path through the application of Prim and Floyd-Warshall algorithms in the year of 2018. The Prim algorithm is widely employed for determining the minimum spanning tree in a weighted graph $G(V, E)$, while the Floyd-Warshall algorithm is commonly utilized to find the shortest paths among all pairs of vertices in a weighted and directed graph. The paper entails a detailed comparative investigation of the Floyd-Warshall and Prim algorithms, utilizing a set of 10 sample graphs.
Anu Pradhan and G. Mahinthakumar, [5] have discussed the parallel implementations and conducted performance analyses of two widely recognized graph algorithms, namely Floyd�Warshall and Dijkstra in 2013. These algorithms are employed for determining the all-pairs shortest path in a large-scale transportation network. The paper also encompassed the derivation of computational time for various parallel implementations of these two graph algorithms. Subsequently, Utti Marina Rifanti and Bongga Arifwidodo [6] have studied the practical implementation of the algorithm in addressing the shortest path problem to explore the optimal route connecting tourism attractions, facilitating increased tourist visits in Banyumas Regency, Indonesia.

Vidhya kannan et al [13] studied of fuzzy Floyd Warshall algorithm and the fuzzy rectangular algorithm to find the shortest path. Broumi, Said et al [7] made an Efficient Approach for Solving Time-Dependent Shortest Path Problem under Fermatean Neutrosophic Environmen. Vidhya, K et al [8] proposed a Novel Method for Finding the Shortest Path With Two Objectives Under Trapezoidal Intuitionistic Fuzzy Arc Costs. Vidhya, K. et al [12] investigated A* search algorithm for the shortest path under interval-valued Pythagorean fuzzy environment. Nedumaran et al [9] developed a comparative study for finding the critical path using triangular fuzzy numbers.

All pairs shortest path Algorithm

The Floyd-Warshall algorithm, coined after Robert Floyd and Stephen Warshall [3], stands as a crucial algorithm in the realms of computer science and graph theory. Its primary purpose is to determine the shortest paths connecting all pairs of nodes within a weighted graph. This algorithm exhibits notable efficiency and versatility, capable of addressing graphs featuring both positive and negative edge weights. Consequently, it proves to be a powerful tool for resolving diverse network and connectivity challenges.

It has quickly become most used approaches for building complicated control systems today.The reason it is suitable for such uses because it closely recycles human decisions and capacity to develop exact solutions from limited ambiguous data.

Definition 2.1 Fuzzy Logic: Fuzzy logic is a method of describing and processing ambiguous data. In the more traditional propositional logic, such as 'it will rain tomorrow', must be either true or false. However, much of the fact's humans use about the world has some ambiguity.

Fuzzy- "Not clear, distinct, or precise; blurred."

Definition 2.2 Fuzzy Set:

If X is an universal set and x∈X, then a fuzzy set $\tilde{A}$ defined as a collection of ordered pairs,

$\tilde{A}$ = {(x, $\mu_{\tilde{A}}$(x)), x ϵ X}

Where $\mu_{\tilde{A}}$(x) is called the membership function that maps X to the membership space M.

Definition 2.3 Triangular Fuzzy number:

A triangular fuzzy number Ã= (a_1, a_2, a_3) is defined by its membership function

$$\mu_A(x) = \begin{cases} 0, & for\ x < a_1 \\ \dfrac{x-a_1}{a_2-a_1} & for\ a_1 \le x \le a_2 \\ \dfrac{a_3-x}{a_3-a_2} & for\ a_2 \le x \le a_3 \\ 0 & for\ x > a_3 \end{cases}$$

Where a_2 denotes the modal value of fuzzy number a_1, a_3 are left and right-hand deviation from the modal or middle value.

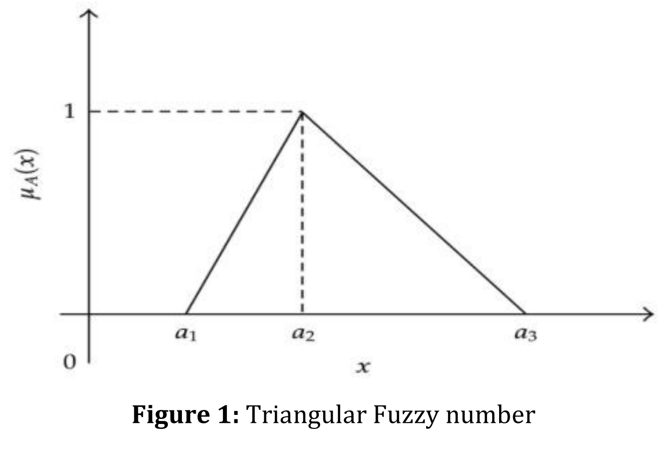

Figure 1: Triangular Fuzzy number

Definition 2.4 Fuzzy Number

Fuzzy set defined on the set R of real numbers is called fuzzy number whose membership function is of the form Ã: R→ [0,1] under certain condition

1. Ã is normal
2. Ã is convex
3. Ã is piecewise continuous

Definition 2.5 Arithmetic Operation Of Triangular Fuzzy Number

For arbitrary triangular fuzzy numbers $\tilde{A} = (a_1, a_2, a_3)$ and $\tilde{B} = (b_1, b_2, b_3)$ and $* = \{+, -, \times, \div\}$,

It is defined by $\tilde{A} * \tilde{B} = \{a_i * b_j / a_i \in \tilde{A}, b_j \in \tilde{B}\}$.

we define

(i)Addition (+): $A + B = (a_1 + b_1, a_2 + b_2, a_3 + b_3)$

(ii)Subtraction (-): $A - B = (a_1 - b_1, a_2 - b_2, a_3 - b_3)$

(iii)Multiplication ($\otimes$) :

$$k \otimes A = (ka_1, ka_2, ka_3), \ k \in R, \ k \geq 0$$

$$A \otimes b = (a_1 b, a_2 b, a_3 b), \ a_1 \geq 0, \ a_2 \geq 0, a_3 \geq 0$$

(iv) Division ($\varnothing$)

$$(A)^{-1} = (a_1, b_1, c_1)^{-1} \cong \left(\frac{1}{c_1}, \frac{1}{b_1}, \frac{1}{a_1} \right), \ a_1 > 0, \ b_1 > 0, \ c_1 > 0$$

$$A \varnothing B \cong \left(\frac{a_1}{c_2}, \frac{b_1}{b_2}, \frac{c_1}{a_2} \right), \ a_1 \geq 0, \ a_2 > 0$$

Definition 2.6 Floyd-Warshall algorithm

- Floyd-Warshall algorithm is a shortest path algorithm that can solve the single- source shortest path problem for a graph with positive or negative edge weights. This algorithm calculates the smallest weights of all paths connecting a pair of vertices and does so at the same time for all pairs of vertices. Robert W. Floyd found this algorithm in 1967. [3] If αβ is the weight from $m\alpha$ to $m\beta$, the potential scenarios are as follows.
- $d_{\alpha\beta} = 0, if \alpha = \beta$
- $d_{\alpha\beta} = \infty, \alpha \neq \beta$ indicates there is no edge from node n_α ton_β.
- $d_{\alpha\beta} = 1, \alpha \neq \beta$ indicates there is edge from node n_α ton_β.

Subsequently, the Floyd-Warshall Algorithm is applied to identify the optimal route. In this context, a matrix of size mtimesm (where $'m'$ represents the number of edges) serves as the graphical representation.$W^0_{\alpha\beta}$denote an adjacency matrix of size mtimesm and $W^r_{\alpha\beta}$represent the shortest path from$n_\alpha$ton_β.

$W^r_{\alpha\beta} = min[W^{r-1}_{\alpha\beta}, W^{r-1}_{\alpha\gamma} + W^{r-1}_{\gamma\beta}]$

METHODOLOGY

This study aims to analyze the tourism growth rate in the state of Tamil Nadu by determining the optimal routes connecting tourist destinations, regardless of various available routes. The tourist places were organized in a graphical representation, wherein nodes represented the various tourist spots in Tamil Nadu, and edges symbolized the paths connecting these spots. The selection of tourist spots was based on data obtained from the Commissioner of Tourism, considering the annual visitation figures for both foreign and domestic tourists.

Algorithm 1 Floyd-Warshall Algorithm

procedure FloydWarshall(d, m)

Initialize w such that $w[\alpha,\beta] = \begin{cases} d[\alpha,\beta] & \text{if there exists an edge from } \alpha \text{ to } \beta \\ \infty & \text{otherwise} \end{cases}$

    for $\gamma = 1$ to m **do**

        for $\alpha = 1$ to m **do**

            for $\beta = 1$ to m **do**

                if $w[\alpha,\gamma] + w[\gamma,\beta] < w[\alpha,\beta]$ **then**

                    $w[\alpha,\beta] = w[\alpha,\gamma] + w[\gamma,\beta]$

                else

                $w[\alpha,\beta] = d[\alpha,\beta]$

tourist spots was based on data obtained from the Commissioner of Tourism, considering the annual visitation figures for both foreign and domestic tourists.

The subsequent phase involves determining edge weights, which will be computed using the normalization method. These edge weights fall under three attributes: distance between the spots, time taken to travel, and the cost of travel between each spot. Data on distance and travel time between each spot can be gathered from Google Maps. The cost of travel between each spot is calculated based on the petrol expenses of the vehicle (car), involving the computation of the car's mileage with the distance (in kilometers) between the spots. The list of data are reduced by allocating Membership functions each by the method of Normalization. Normalization of big data can also be done by using MS Excel by taking maximum and minimum values. The nodes connected by edges signify direct connections without overlapping with other nodes.

2.7 Graphical Representation

The list of Tourists spots and their connections are represented by the graph, consists of 12 vertices(Tourists spots) and the edges(path). The Edge weight of the graph can be calculated basedon the attributes such as Distance, Time and Travel Expenses.

The list of vertices in the graph are: 0-Chennai, 1-Kancheepuram, 2-Cuddalore, 3-Trichy, 4- Thanjavur, 5-Nagapattinam, 6-Rameshwaram, 7-Kanyakumari, 8-Madurai, 9-Kodaikanal, 10- Coimbatore, 11-Ooty.

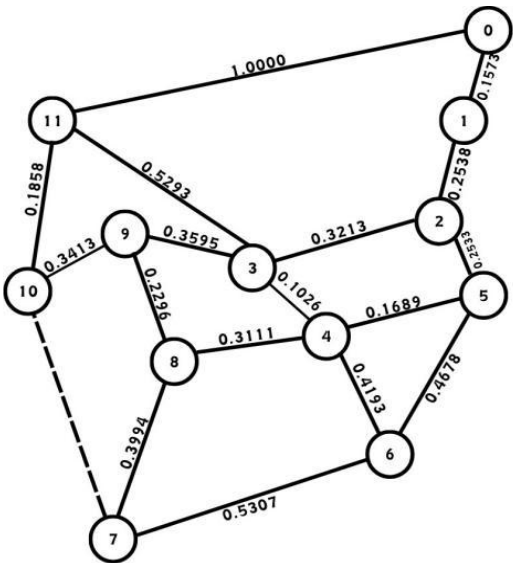

Figure 2: Graphical Representation of the collected data

The list of data are as follows:

Table 1: Attributes data collection

Path	Distance(km)	Time(min)	Travel Expenses(rs)
Chennai-Kancheepuram	75	139	398.66
Chennai-Ooty	558	684	2966.05
Kancheepuram-Cuddalore	137	185	728.22
Cuddalore-Trichy	187	201	994.00
Trichy-Thanjavur	57	71	302.98
Cuddalore-Nagapattinam	126	211	669.75
Nagapattinam-Rameshwaram	266	308	1413.92
Rameshwaram-Kanyakumari	308	334	1637.17
Kanyakumari-Madurai	243	224	1291.67
Madurai-Kodaikanal	116	187	616.60
Trichy-Kodaikanal	201	245	1068.42
Thanjavur-Madurai	185	185	983.37
Kodaikanal-Coimbatore	176	269	935.53
Coimbatore-Ooty	85	173	451.82
Trichy-Ooty	284	390	1509.60
Thanjavur-Nagapattinam	88	131	467.76
Thanjavur-Rameshwaram	247	255	1312.93

3. Calculation

The resulting list of nodes and their respective edge weights after normalization are as follows:

Table 2: List of Places with direct edge weights

S.No	Places	Weights
1	Chennai- Kancheepuram	0.1573
2	Chennai- Ooty	1.0000
3	Kancheepuram- Cuddalore	0.2538
4	Cuddalore - Trichy	0.3213
5	Trichy- Tanjavur	0.1026
6	Cuddalore- Nagapattinam	0.2533
7	Nagapattinam- Rameshwaram	0.4678
8	Rameshwaram- Kanyakumari	0.5307
9	Kanyakumari- Madurai	0.3994
10	Madurai- Kodaikanal	0.2296
11	Trichy- Kodaikanal	0.3595
12	Tanjavur- Madurai	0.3111
13	Kodaikanal- Coimbatore	0.3413
14	Coimbatore- Ooty	0.1858
15	Trichy- Ooty	0.5293
16	Thanjavur- Nagapattinam	0.1689
17	Thanjavur- Rameshwaram	0.4193

The initial adjacency matrix can represent the list of nodes along with their edges, without the inclusion of any intermediary nodes. According to the algorithm, if there is a direct edge between nodes, the corresponding edge weights are considered. If there is an edge-to-edge pattern (with the same edge), a value of 0 is assigned. In cases where there is no direct adjacency edge between nodes, the value of infinity is assigned. Based on these conditions, the initial adjacency matrix is createdfor the 12 vertices listed in the rows and column wise.

Table 3: $\gamma = 0$(ie,NO Intermediate nodes)

Vertices	0	1	2	3	4	5	6	7	8	9	10	11
0	0	0.1573	∞	∞	∞	∞	∞	∞	∞	∞	∞	1.0000
1	0.1573	0	0.2538	∞	∞	∞	∞	∞	∞	∞	∞	∞
2	∞	0.2538	0	0.3213	∞	0.2533	∞	∞	∞	∞	∞	∞
3	∞	∞	0.3213	0	0.1026	∞	∞	∞	∞	0.3595	∞	0.5293
4	∞	∞	∞	0.1026	0	0.1689	0.4193	∞	0.3111	∞	∞	∞
5	∞	∞	0.2533	∞	0.1689	0	0.4678	∞	∞	∞	∞	∞
6	∞	∞	∞	∞	0.4193	0.4678	0	0.5307	∞	∞	∞	∞
7	∞	∞	∞	∞	∞	∞	0.5307	0	0.3994	∞	∞	∞
8	∞	∞	∞	∞	0.3111	∞	∞	0.3994	0	0.2296	∞	∞
9	∞	∞	∞	0.3595	∞	∞	∞	∞	0.2296	0	0.3413	∞
10	∞	∞	∞	∞	∞	∞	∞	∞	∞	0.3413	0	0.1858
11	1.0000	∞	∞	0.5293	∞	∞	∞	∞	∞	∞	0.1858	0

The initial adjacency matrix is then undergone the procedure of algorithm by keeping γ as 1. Keeping (0,0) as a fixed values, the other columns of the matrix are proceeded.
$d[11,1] = min\{d[11,1], d[11,0] + d[0,1]\} = min\{\infty, 1.1573\} = 1.1573$

Table 4:$\gamma = 1$(ie, Using {0} as an intermediate node)

Vertices	0	1	2	3	4	5	6	7	8	9	10	11
0	0	0.1573	∞	∞	∞	∞	∞	∞	∞	∞	∞	1.0000
1	0.1573	0	0.2538	∞	∞	∞	∞	∞	∞	∞	∞	**1.1573**
2	∞	0.2538	0	0.3213	∞	0.2533	∞	∞	∞	∞	∞	∞
3	∞	∞	0.3213	0	0.1026	∞	∞	∞	∞	0.3595	∞	0.5293
4	∞	∞	∞	0.1026	0	0.1689	0.4193	∞	0.3111	∞	∞	∞
5	∞	∞	0.2533	∞	0.1689	0	0.4678	∞	∞	∞	∞	∞
6	∞	∞	∞	∞	0.4193	0.4678	0	0.5307	∞	∞	∞	∞
7	∞	∞	∞	∞	∞	∞	0.5307	0	0.3994	∞	∞	∞
8	∞	∞	∞	∞	0.3111	∞	∞	0.3994	0	0.2296	∞	∞
9	∞	∞	∞	0.3595	∞	∞	∞	∞	0.2296	0	0.3413	∞
10	∞	∞	∞	∞	∞	∞	∞	∞	∞	0.3413	0	0.1858
11	1.0000	**1.1573**	∞	0.5293	∞	∞	∞	∞	∞	∞	0.1858	0

Repeating the process for γ = 12, the output matrix D^{12} is obtained.

Table 5: γ=12(ie, Using {0,1,2,3,4,5,6,7,8,9,10,11} as an intermediate nodes)

Vertices	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
0	0	0.1573	0.4111	0.7324	0.8333	0.6644	1.1322	1.5438	1.1444	1.0919	1.1858	1.0000
1	0.1573	0	0.2538	0.5751	0.6760	0.5071	0.9749	1.3865	0.9871	0.9346	1.2759	1.1044
2	0.4111	0.2538	0	0.3213	0.4222	0.2533	0.7211	1.2518	0.7333	0.6808	1.0221	0.8506
3	0.7324	0.5751	0.3213	0	0.1026	0.2715	0.5219	0.8131	0.4137	0.3595	0.7008	0.5293
4	0.8333	0.6760	0.4222	0.1026	0	0.1689	0.4193	0.7105	0.3111	0.4621	0.8034	0.6319
5	0.6644	0.5071	0.2533	0.2715	0.1689	0	0.4678	0.8794	0.4800	0.6310	0.9723	0.8008
6	1.1322	0.9749	0.7211	0.5219	0.4193	0.4678	0	0.5307	0.7304	0.8814	1.2227	1.0512
7	1.5438	1.3865	1.2518	0.8131	0.7105	0.8794	0.5307	0	0.3994	0.6290	0.9703	1.1561

8	1.1444	0.9871	0.7333	0.4137	0.3111	0.4800	0.7304	0.3994	0	0.2296	0.5709	0.7567
9	1.0919	0.9346	0.6808	0.3595	0.4621	0.6310	0.8814	0.6290	0.2296	0	0.3413	0.5271
10	1.1858	1.2759	1.0221	0.7008	0.8034	0.9723	1.2227	0.9703	0.5709	0.3413	0	0.1858
11	1.0000	1.1044	0.8506	0.5293	0.6319	0.8008	1.0512	1.1561	0.7567	0.5271	0.1858	0

4. Results from Output matrix

This is an output matrix which gives the minimum weight between all pairs of vertices (i.e, from 0to 11). Since the diagonal elements are 0, there is no path from the vertex itself. From this Output matrix, the path between all pairs of vertices is found.

Here are the list of nearest tourists spots from the output matrix:

- The nearest spot from Chennai is Kancheepuram with approximately 8170821 visitors per year and the weight is 0.1573
- The nearest spot from Cuddalore is Nagapattinam with approximately 1961500 visitors per year and the weight is 0.2533
- The nearest spot from Trichy is Thanjavur with approximately 6328151 visitors per year and the weight is 0.1026
- The nearest spot from Trichy, Nagapattinam and Rameshwaram is Thanjavur with approximately 6328151 visitors per year and the weight is 0.1026,0.1689,0.4193
- The nearest spot from Kanyakumari and Kodaikanal is Madurai with approximately 13958585 visitors per year and the weight is 0.3994,0.2296
- The nearest spot from Coimbatore is Ooty with approximately 2930919 visitors per year and the minimum weight is 0.1858

4.1 Results

Based on the Output matrix, the most optimum transportation route that also covers the majortoruists spots in Tamil Nadu are given below.

The list of places consists of 4 major cities(i.e, Chennai, Coimbatore, Madurai, Trichy).

The Optimum route from Vertex 0 (Chennai) to other vertices are:

Table 6: Result:1

Vertices	Tourists spots	Route	Weight
0-2	Chennai-Cuddalore	0-1-2	0.4111
0-3	Chennai-Trichy	0-1-2-3	0.7324
0-5	Chennai-Nagapattinam	0-1-2-5	0.6644
0-4	Chennai-Thanjavur	0-1-2-5-4	0.8333
0-6	Chennai-Rameshwaram	0-1-2-5-6	1.1322
0-9	Chennai-Kodaikanal	0-1-2-3-9	1.0919
0-7	Chennai-Kanyakumari	0-1-2-5-4-8-7	1.5438

The Optimum route from Vertex 3 (Trichy) to other vertices are:

Table 7: Result:2

Vertices	Tourists spots	Route	Weight
3-7	Trichy-Kanyakumari	3-9-8-7	0.9855
3-6	Trichy-Rameshwaram	3-4-5-6	0.7393
3-10	Trichy-Coimbatore	3-9-10	0.7008

The Optimum route from Vertex 8 (Madurai) to other vertices are:

Table 8: Result:3

Vertices	Tourists spots	Route	Weight
8-5	Madurai-Nagapattinam	8-4-5	0.4800
8-1	Madurai-Kancheepuram	8-4-5-2-1	0.9871
8-11	Madurai-Ooty	8-9-10-11	0.7567
8-6	Madurai-Rameshwaram	8-4-6	0.7304

The Optimum route from Vertex 7 (Kanyakumari) to other vertices are:

Table 9: Result:4

Vertices	Tourists spots	Route	Weight
7-11	Kanyakumari-Ooty	7-8-9-10-11	1.1561

REFERENCES

[1] Harsha Pandey and PP Pande. Applications of algorithmic graph theory to the real world problems. Int. J. Innov. Sci. Res., 10(2):303–307, 2014.
[2] Edsger W Dijkstra. A note on two problems in connexion with graphs. In Edsger Wybe Dijkstra:His Life, Work, and Legacy, pages 287–290. 2022.
[3] Robert W Floyd. Algorithm 97: shortest path. Communications of the ACM, 5(6):345–345, 1962.
[4] Andysah Putera Utama Siahaan and Mesran Mesran. Prim and floyd-warshall comparative algorithms in shortest path problem. 2018.
[5] Anu Pradhan and G Mahinthakumar. Finding all-pairs shortest path for a large-scale transportation network using parallel floyd-warshall and parallel dijkstra algorithms. Journal of computing in civil engineering, 27(3):263–273, 2013.
[6] Utti Marina Rifanti and Bongga Arifwidodo. Tourism transportation's optimum route in banyu mas using the weighted fuzzy graph. In AIP Conference Proceedings, volume 2194. AIP Publishing, 2019.
[7] Vidhya.K, Saraswathi. A. An Efficient Approach for Solving Time-Dependent Shortest Path Problem under Fermatean Neutrosophic Environment, Neutrosophic sets and Systems,82-94, 2024.
[8] Vidhya.K, Saraswathi.A. A Novel Method for Solving the Time-Dependent Shortest Path Problem under Bipolar Neutrosophic Fuzzy Arc Values, Neutrosophic sets and Systems, 80-100, 2024.
[9] Saraswathi.A, P.Nedumaran. Comparative Study to Find the Critical Path Using Triangular Fuzzy Number". Journal of Computational Analysis and Applications, 33(5), 345-354, 2024.
[10] Vidhya.K, Saraswathi.A. A Novel Method for Finding the Shortest Path With Two Objectives Under Trapezoidal Intuitionistic Fuzzy Arc Costs", International Journal of Analysis and Applications, 21, pp 1-15, 2023.
[11] Vidhya.K, Saraswathi.A. "Employing the Bellman-Ford Algorithm with Score Functions to Address the Linear Diophantine Fuzzy Shortest Path Problem in Network Analysis", Mathematical Modelling of Engineering Problems, pp 1884-1892, 2023.
[12] Vidhya.K, Saraswathi.A.An Improved A∗ Search Algorithm for the Shortest Path under Interval Valued Pythagorean Fuzzy Environment, granular computing, 8(2), pp 1-11, 2022.
[13] Vidhya.K, Saraswathi.A. Comparative Study of Fuzzy Floyd Warshall Algorithm and the Fuzzy Rectangular Algorithm to find the shortest path", AIP Conference Proceedings, 2516 (1), pp 1-9, 2022.