*International Journal of Computer Systems (ISSN: 2394-1065), Volume 02– Issue 06, June, 2015*
*Available at http://www.ijcsonline.com/*

# Effect Self Loop Avoidance Weighted Non Deterministic Finite Automatic SDT Based SpeedoCounter Comparison Algorithm for Efficient and Congested Free Path in Travelling Environments

Lakshmi Kartheek V [A], Raghuram Ch [B]

[A] Computer Science & Eng. Dept., S.R.K.R Engineering College, Bhimavaram, Andhrapradesh, India.

[B] Computer Science & Eng. Dept., Andhra University, Visakhapatnam, Andhrapradesh, India.

***Abstract***

*Nowadays in computer networks or travelling networks, the routing of vehicles or moving objects is mainly based on the best path. This will help in reducing the costs of paths in a place, city or any specified region. But, at the same time every shortest path need not be an efficient path. So, in this paper, we proposed an algorithm called Loop Avoidance Weighted Non Deterministic Finite Automatic SDT Based Counter Comparison Algorithm and applied on various networks. The main objective of this paper is to find the efficient and congested free path using three parameters speed, distance and time. An example is illustrated for finding optimal and efficient solutions to travelling is presented. The practical results show that the proposed method obtains better and accurate paths when compared with the existing methods.*

***Keywords:*** *Computer Networks, Routing, Shortest Path, SDT, Counter, NDFA.*

## I. INTRODUCTION

In the past, the shortest path can be found by mostly considering the weights [1]. The shortest path problem is used to finding the shortest path or route from a starting point to a final destination. Generally, in order to represent the travelling environments we use graphs. [2] A graph is like a network, which contains some sets of vertices and edges. Edges connect to pairs of vertices. Along the edges of a graph, it is possible to travel by moving from one vertex to other vertices [3]. Depending on one can walk along the edges by both sides or by only one side determines if the graph is a directed graph or an undirected graph [4].In addition, lengths of edges are often called weights, and the weights are generally used for calculating the shortest distance from one point to another point. [5] In the real world it is possible to apply the graph theory to Different types of scenarios. For example, in order to represent a network we can use a graph, where vertices represent places and edges represent routes that connect the cities [6]. If routes are one-way then the graph will be directed; otherwise, it will be undirected. There exist different types of algorithms and methods that solve the shortest path problem.

The use of shortest paths or best and accurate paths occurs naturally when travelling between two points, whether this is from one place to another, that is, from one street to another, or from one city to another.[7] Taking a long path typically makes no sense, since doing so results in time being wasted. But, all shortest paths need not be an efficient path. To achieve the greatest efficiency when travelling between two points, it is necessary to take a path that is shortest and efficient among all possible paths; But along with the shortest distance, the speed and time of a vehicle in that path.

The problem of computing shortest path distance commonly arises when the most cost-efficient route through a travel network needs to be found. [8] In the case of transportation, cost may be represented by a number of factors, including distance, time spent, or many other important factors.

## II. BACK GROUND AND RELATED WORK

The different algorithms were proposed for shortest path in a network. [9] But most of the algorithms taking care about only distance but not traffic conditions in a particular path. The truly shortest path, or that of minimum cost, is not always be the accurate path. For example, consider finding the minimum distance and efficient path in order to minimize the time spent travelling between two locations in a city. Here cost is measured in terms of the time spent travelling. [10] In order to avoid traffic congestion, we must take care of some other important factors. Such a type of path can be completely different from the path that is shortest in terms of distance travelled. Furthermore, large shortest path problems are typically too complex to solve accurately but it is possible with counter summation. [11] By computing shortest paths, rather than not only measure the distance, accurate result can always be obtained. The edges in an undirected graph have no direction, and can be thought of as allowing travel in both directions.

In contrast, the edges in a directed graph have an associated direction, which can be thought of as specifying the direction of travel. [12] The edges in a directed graph as being one-way, and edges in an undirected graph as two-way. [13]The edges of a graph of networks can be weighted, in which case each edge has an associated cost. In the case of a vehicle transportation network, this cost may be the distance along a road between two vertices. Shortest path problems are represented using directed graphs, since the cost from one vertex to another may be different in the opposite direction. [1]The edges in a graph represent paths connecting vertices. Any such path similarly has an associated cost of distance, corresponds to the sum of costs of edges along the path.

### A. *Dijkstra's Algorithm: Explanation*

In a network, for each vertex within a graph we assign a label that determines the minimal length from the starting point s to other vertices v of the graph. [3] The algorithm works sequentially, and in each step it tries to decrease the value of the label of the vertices. The algorithm stops when all vertices have been visited. The label at the starting point s is equal to zero (d[s]=0); however, labels in other vertices v are equal to infinity which means that the length from the starting point s to other vertices is unknown. In addition, for each vertex v we have to identify whether it has been visited or not. Initially, all vertices are assigned as unvisited (u[v] = F). [3] The Dijkstra's algorithm consists of n iterations. If all vertices have been visited, then the algorithm finishes; otherwise, from the list of unvisited vertices we have to choose the vertex which has the minimum (smallest) value at its label.

### B. *Floyd-Warshall Algorithm: Explanation*

Consider the graph G, where vertices were numbered from 1 to n. [3]The notation dijk means the shortest path from i to j, which also passes through vertex k. obviously if there exists an edge between vertices i and j it will be equal to dij0, otherwise it can assigned as infinity. However, for other values of dijk there can be two choices: If the shortest path from i to j does not pass through the vertex k then value of dijk will be equal to dijk-1. If the shortest path from i to j passes through the vertex k then first it goes from i to k, after that goes from k to j.

### C. *Bellman-Ford Algorithm: Explanation*

In comparison to Dijkstra's algorithm, the Bellman-Ford algorithm admits the edges with negative weights. [3] that is why, a graph can contain cycles of negative weights, which will generate numerous number of paths from the starting point to the final destination, where each cycle will minimize the distance weights of the shortest path. The array will store the minimal length from the starting point s to other vertices. Values a and b are vertices of the graph, and c is the corresponding edge that connects them.

## III. PROPOSED WORK

In this wok, we are using Non deterministic finite automata. From this NDFA, we eliminate the loops, so that we can get the paths from source to destination without having any ambiguity. More precisely, we propose a new Loop Avoidance Weighted Non Deterministic Finite Automatic SpeedoCounter Comparison Algorithm which does not require any restriction or difficulty. This LANDFASSCC algorithm is applied on the Networks and getting Accurate shortest paths among various possibilities. In this LANDFASCC algorithm, we are using a counter for Summing Weights of edges on a network graph. Then we add this counter values to times taken in that path.

### A. *Weighted Non Deterministic Finite Automata*

We proposed Non deterministic finite automata with weighted edges. It is a method that tries to assign weights to edges of NDFA. The input weights of paths can be assigned and applied our LANDFASSCC algorithm. In this NDFA, for the same input symbol, it will traverse to two or more paths. So, we assign the weights to input symbols. This gives a chance to getting maximum likelihood of getting best results. Its main role is to calculate the counter values for each path in the network. It is an important method, which is mainly used to finding the minimum weighted counter path along with minimum time of the network model.

The following shows a pictorial representation of a NDFA with weights assigned for the input variables of the network graph.

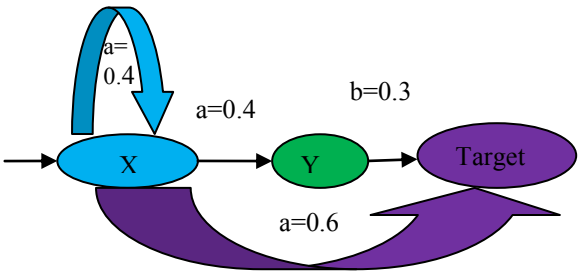


Fig 1: Weighted Non Deterministic Finite Automata

### B. *Loop Avoidance NDFA Method*

This method is proposed to ignore the loops on the network graph. Then find out the required weighted counter values and times taken towards in a particular path.

### C. *Counter Comparison*

The counter value will be calculated for each path in a network graph. Then the Comparison is takes place among the counter values of multiple paths. The minimum counter value of a path will be the shortest path among multiple paths in a network graph.

### *D. LANDFASSCC Algorithm*

Our proposed Loop Avoidance Weighted Non Deterministic Finite Automatic Speedo Counter Comparison Algorithm is used to find out the minimum time taken by a shortest path among the multiple paths in network graph.

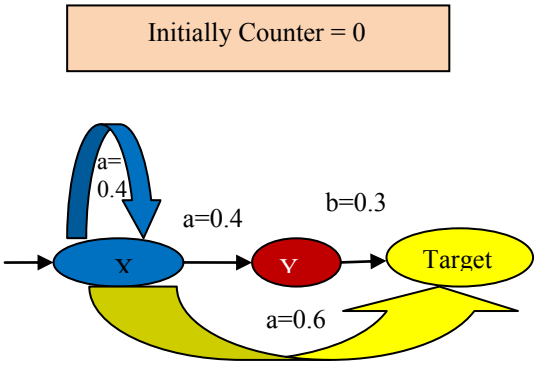


Fig 2: Weighted NDFA with Counter

Here we are taking a counter which is taking initially as zero. After that, the weighted value of edges can be added to each counter. The number of counters will be formed based on the number of possibilities of paths from source to destination. The each counter values can be summed to times taken in that path. Then all the Speedocounter values can be compared and gives the smallest distance with less time among the possibilities. The main purpose of using this method is to find out the minimal path for best routing.

$$\text{Counter}_{P_i} = w_{i1} + w_{i2} + w_{i3+} \ldots\ldots\ldots + w_{iD}$$

$1 <= i <= \text{Dest}$ for Each path "$P_i$"

---------- (1)

$$\text{Speedocounter}(P_i) = [\text{Counter}_{pi}] + \frac{(Tpin - Tpi1) * 100}{Hours\ per\ day}$$

---------- (2)

Where

$P_i$ means Paths possible from source to destination.

$< i = 1, 2, 3, \ldots\ldots n; >$

$T_{pi1}$ means starting time of various paths available from source to destination.

$T_{pin}$ means destination time of various paths available from source to destination.

### *E. Proposed LANDFASSCC Algorithm*

Our proposed Loop Avoidance Weighted Non Deterministic Finite Automatic Speedo Counter Comparison algorithm is developed using three parameters speed, distance and time. In this algorim, we are maintaining a Counter, and also developed a formula to find out the times taken by various paths in the network.

Initialization:

Initially every counter value : counter = 0
Starting Location Time for each path : $T_1$
Destination Location Time for each path: $T_{pin}$

Input:

$W_i = <w_{i1}, w_{i2}\ w_{i3} \ldots\ldots\ldots\ w_{iD}>$
& $1 <= i <= \text{Dest}$
T1, Tn

Output:
Efficient & Congested free Path "$P_i$" is obtained

Procedure for LANDFASSCC Algorithm:

1. Take a NDFA, and identify all possible paths from source to destination.
2. Avoid the Self loops in a particular NDFA.
3. Finding the sum of weights for each path from source to destination Using Eq (1).
4. Note down the starting and destination location time of a moving object in each path and substitute in Eq (2).
5. By Equation (2), we will get the Speedocounter values for each possible path from Source to destination.
6. We will compare the speedocounter values for all the paths available from source to destination.
7. Finally, the path which has Minimum Speedocounter value will be the required efficient and congested free path.
8. Return the congested free path $P_i$;

End Procedure.

## IV. EXPERIMENTAL RESULT

In our experiment, we are taking a NDFA, and the weights are also assigned. The self loops can be avoided. By Using the Counter formula we are calculating the sum of weights for each path on the network. From this, we will find out the minimum weighted path from source to destination. But all shortest paths need not be convenient.

Then we will calculate the time it will take to destination from source using speedocounter equation. Finally we compare the speedocounter values. The path which has minimum speedocounter value will be the convenient path.

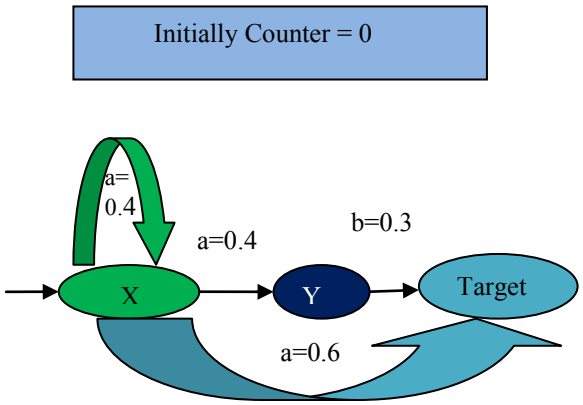


Fig 3: Weighted NDFA Network with Counter value

TABLE I: Available paths and their Weights

| S.NO | SOURCE | DEST | PATH | WEIGHTS |
|---|---|---|---|---|
| 1. | X | Target | X->y->Target | 0.4,0.3 |
| 2. | X | Target | X->Target | 0.6 |

TABLE II: Starting Time ($T_1$) and Destination Time ($T_n$)

| S.NO | SOURCE | DEST | PATH | StarttimeT1 DstTimeTn |
|---|---|---|---|---|
| 1. | X | Target | X->y->Target | 1,4 |
| 2. | X | Target | X->Target | 1,5 |

Suppose, there are two paths in the above network. These are X->y->Target and X->Target. The weights of these paths are illustrated in Table 1. The Starting time (T1) and Destination Time (Tn) are illustrated in Table2.

By Using Equation (1), the counter value can be calculated for all possible paths.

For Path $P_1$,

Counter = 0.4+0.3 = 0.7

For Path $P_2$,

Counter = 0.6

From, this we can observe that Path $P_2$ be the Shortest Path based on the Distance. But all shortest paths need not be a convenient and congested free path.

So, we can find out the path which has more accuracy in terms of speed, distance and time. By using Equation (2), we can find the following values.

For path $P_1$,

$$\text{Speedocounter } (P_1) = 0.7 + \frac{(4-1)*100}{24} = 13.2$$

For Path $P_2$

$$\text{Speedocounter } (P_2) = 0.6 + \frac{(5-1)*100}{24} = 17.2$$

Compare these speedocounter values of paths p1 and p2.

i.e,

$$\text{Speedocounter } (P_1) < \text{Speedocounter } (P_2)$$

So, the path $P_1$ has less speedocounter value than path $P_2$. So from this we get the minimum and congested free path $P_1$.

Finally, from this we understand that even path2 has minimum weight in terms of distance; it is not an accurate path to go. So, in this paper, we consider not only distance but also speed and other traffic conditions in a particular route.

Thus we will get the most convenient and congested free path $P_1$ in terms of speed, distance and time. This can be useful for Travelling in a right path. It is consistent in accuracy and finding the right and useful path to journey to a destination.

TABLE 3: Path Selection Using Landfasscc Algorithm

| Input | Best Path Illustration (%) | |
|---|---|---|
| | *Speedcounter value* | *LANDFASSCC Algorithm* |
| Path 1 | 13.2 | Using Speedcounter formulas 1 and 2 |
| Path 2 | 17.2 | Using Speedcounter formulas 1 and 2 |
| Speedcounter (P1) < Speedcounter(P2) | | |
| RESULT : Path P1 is the best path | | |

In this tables, we are illustrated a detailed comparison Between two paths speedcounter values.

Finally we identified the best path among various Paths in a given network graph. In this way, we will get accurate and efficient path of given network.

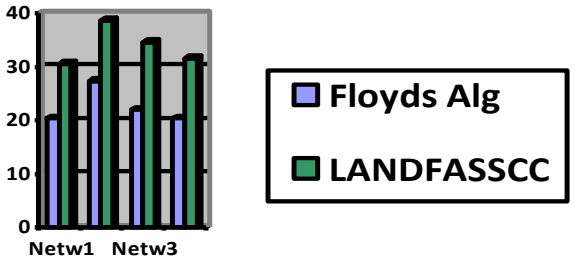


Fig 4: Graphical representation Of Comparison

From this graphical representation, we know that the LANDFASSCC algorithm performs well when compared to the other algorithm like Floyds algorithm. So, the proposed algorithm performing well not only in the context of distance but also with speed, distance and time.

## V. CONCLUSION & FUTURE WORK

In this Work, Loop Avoidance Weighted Non Deterministic Finite Automatic SpeedoCounter Comparison Algorithm is proposed and applied to different Travelling networks. This algorithm gives the accurate and convenient path not only in terms of Distance but also speed and time. It gives the More Accurate and efficient Results by considering all the situations when compared to Existing System. Thus we found maximum likelihood estimates of getting best results. In future, research work it is possible to implement by considering weather conditions.

## ACKNOWLEDGMENT

Our thanks to the experts who have contributed towards development of this paper.

## REFERENCES

[1] Routray, S.K.; Sahin, G.; da Rocha, J.R.F.; Pinto, A.N. "Statical Analysis and Modeling of Shortest PathLengths in Optical Transport Networks", 2015, Volume:33,IEEE.

[2] D. Prangchumpol , "a Network Traffic Prediction Algorithm based On Data Mining Technique" , 2013, Volume 7, world Academy of Science, Engineering and Technology.

[3] Kairanbay Magzhan, Hajar Mat Jani. , "A Review And Evaluations of Shortest Path Algorithms", June 2013, Volume :2, International Journal Of Scientific and Technology Research.

[4] D Dhayalan, K Kanimozhi, " Least cost and accurate estimation of Shortest path in Large graph, 2015,Vol 2, Issue 3.

[5] Mrs. S.P. Gaikwad, Ms. Priyanka Patil," Routing Mechanis for the Improvement of Network Throughput", May 2014, IJARCSSE, Vol4, Issue 5.

[6] Mr. Kuldeep Kumar ," Role of Genetic Algorithm in Routing for Large Networkl et.al International Journal on Computer Science and Engineering (IJCSE).

[7] Nenad S. Kojić1, Marija B. Zajeganović Ivančić1, Irini S.Reljin2 and Branimir D. Reljin2,"New algorithm for Packet routing in mobile ad-hoc networks", Journal Of Automatic Control, University Of Belgrade, 2010, Vol. 20:9-16,.

[8] Mohd Yazid Idris, Abdul Hanan Abdulah, Mohd Aizaini Maarof,- A Synthetic Network Traffic Generator Tool for Normal and Anomaly Traffic Model" 2006.

[9] SARVESH S. KULKARNI, G. R. DATTATREYA" Statistically Multiplexed Adaptive Routing Technique for Ad Hoc Networks Wireless Networks" 2004 , 10, 89–101.

[10] Andrej KOS, Ljubljana, Slovenia, Janez BEŠTER 25, 1000 Ljubljana, Slovenia " Poisson Packet Traffic Generation Based on Empirical Datal.

[11] LIPING FU*. 3G1L. R. RILETT , Texas A&M "Expected Shortest Paths In Dynamic And Stochastic Traffic Networks" Transpn Res., 1998, Vol. 32, No 7.

[12] Sudhanshu Choudhary Sha¯ Qureshi," Implementation of a New Architecture and Routing Algorithm" , International Journal of Automation and Computing 9(4), August 2012, 403.

[13] Tirtharaj Dash, Tanistha Naya ,"Quantum Finite Automata: a Language Acceptor Model ",sept 2012 Volume2, Issue 9, IJARCSSE.