Efficient Data Dissemination for Topology Management in VANETs

Dissertation submitted in partial fulfillment of the requirements for the award of degree

of

Masters of Technology

in

Computer Science and Applications

Submitted By
Abhishek Khanna
(Roll No. 651103002)

Under the supervision of
Dr. Rajesh Kumar
Associate Professor & Head, SMCA

School of Mathematics and Computer Applications
Thapar University
Patiala –147004

July 2014

Certificate

I hereby certify that the work which is presented in the thesis entitled, "**Efficient Data Dissemination for Topology Management in VANETs**", in partial fulfillment of the requirements for the award of degree of the Master in Technology in **Computer Science and Applications**, submitted in School of Mathematics and Computer Applications of Thapar University, Patiala is authentic record of my own work carried out under supervision of Dr. Rajesh Kumar and refers others researcher's work which are duly listed in the reference section.

The matter presented in the thesis has not been submitted for award of any other degree of this or any other university.

Abhishek Khanna
651103002

This is to certify that above statement made by the candidate is correct and true to the best of my knowledge.

Dr. Rajesh Kumar
Associate Professor
School of Mathematics and Computer Applications

Countersigned by:-

Dr. Rajesh Kumar
Head
School of Mathematics and Computer Applications
Thapar University
Patiala

Dr. S.K.Mahopatra
Dean Academic Affairs
Thapar University
Patiala

Acknowledgment

I would like to acknowledge everyone who supported me during my experience at Thapar University and my work on this thesis. First of all I would like to thank my family who stayed with me and supported me all the time while I made this dream come true. I extend my thanks to my thesis supervisor, Dr. Rajesh Kumar, who helped me to materialize my ideas on this thesis. His enthusiasm and optimism made this experience both rewarding and enjoyable. Most of the novel ideas and solutions found in this thesis are the result of our numerous stimulating discussions. His feedback and editorial comments were also valuable for writing this thesis.

My thesis work would be incomplete without thanking my friends who were always there in the hour of need.

Abstract

Recent research, development and standardization advances in vehicular ad hoc networks (VANETs) have motivated increasing interest in various data services for in-vehicle consumption in terms of ‘commerce’ on wheels. These include a wide variety of applications: local information (e.g., traffic notification, map updates, location-based advertisements) pushed to vehicles; or specific data pulled from Internet servers (e.g., neighborhood parking, reviews of local restaurants, and video clips of local attractions). Here the process of Dissemination refers to the field of communication, means to broadcast a message to the public without direct feedback from the audience. Dissemination takes on the theory of traditional view of communication, which involves a sender and a receiver. The traditional communication view point is broken down into a sender sending the information, and receiver collecting the information for further processing.

The second terminology used in the process is Vehicular ad hoc Network (VANETs) that uses cars as mobile nodes to create a mobile network. A VANET turns every participating car into a wireless router or node, allowing cars approximately 100 to 300 meters of each other to connect and thus creating a network with a wide range. As cars fall out of the signal range and drop out of the network, other cars can join in, connecting vehicles to one another so that a mobile internet is created. It is estimated that the first systems that will integrate this technology are police and fire vehicles to communicate with each other for safety purposes. Here the entire process has been built up in an order to make use of VANETs in such a manner that there is assurance of 100% correct data transfer from a source to destination with minimal hopping/shifting of data from one node to another selecting the shortest path using GPS system which is guiding the vehicle in route.

List of Figures

List of Tables

Table of Contents

Chapter 1

Introduction

Introduction

It is very necessary to understand the basic conceptual fundaments an attribute before initiating upon any new operation so that the resultant output may be very clear on the existing system. In regard to the same, we have first tried to understand the basic fundamental terminology of the local key aspects of various terms used in the entire dissertation work. Hence in order to satisfy the same we had given a brief introduction upon the basics of key aspects like Vehicular Ad Hoc Network, its components and architecture, characteristics along with pros and cons of Ad Hoc Network.

1.1 Vehicular Ad Hoc Network

In the modern world of Information Technology perhaps the most easiest and understandable quotation to VANETs can be referred to as, "**Computers on the Wheels**" or "**Networks on the Wheel**s." Vehicular ad hoc networks (VANETs) is a subclass of mobile Ad Hoc networks (MANETs) having a promising approach for future intelligent transportation system (ITS). The direct communication between vehicles using an Ad Hoc network referred to as

- Inter-vehicle communication (IVC)
- Car-to-Car communication(C2CC)

An advantage to use these VANETs is to achieve the following two goals

- Information sharing
- Co-operative driving

Vehicular networks are a novel class of wireless networks that have emerged thanks to advances in wireless technologies and the automotive industry. Vehicular networks are spontaneously formed between moving vehicles equipped with wireless interfaces that could be of homogeneous or heterogeneous technologies. These networks, also known as VANETs, are considered as one of the ad hoc network real-life application enabling communications among nearby vehicles as well as between vehicles and nearby fixed equipment, usually described as roadside equipment [4]. Vehicles can be either private, belonging to individuals or private companies, or public transportation means (e.g., buses and public service vehicles such as police cars). Fixed equipment can belong to the government or private network operators or service providers.

Indeed, vehicular networks are promising in allowing diverse communication services [17] to drivers and passengers. These networks are attracting considerable attention from the research community as well as the automotive industry. High interest for these networks is also shown from governmental authorities and standardization organizations. In this context, dedicated short-range communications (DSRC) system has emerged in North America, where 75 MHz of spectrum was approved by the U.S. FCC (Federal Communication Commission) in 2003 for such type of communication that mainly targets vehicular networks [9]. On the other hand, the Car-to-Car Communication Consortium (C2C-CC) has been initiated in Europe by car manufacturers and automotive OEMs (original equipment manufacturers), with the main objective of increasing road traffic safety and efficiency [8] by means of inter vehicle communication. IEEE is also advancing within the IEEE 1609 family of standards for wireless access in vehicular environments (WAVE).

A modern vehicle which is being used as a VANET may have a following structure

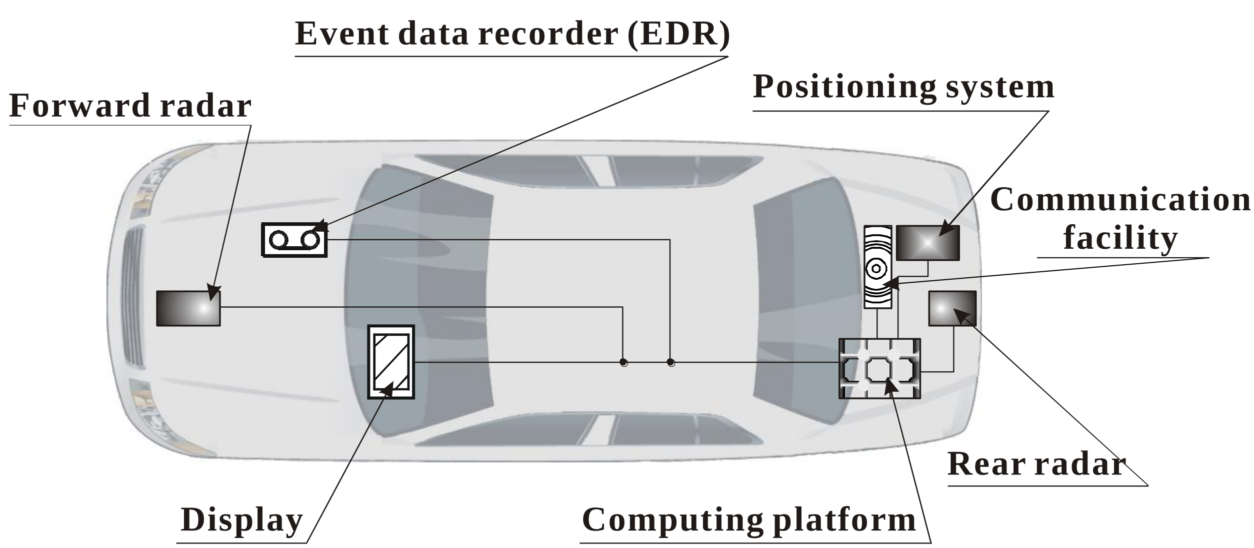


Figure 1.1

1.2 Overview of basic components in VANETs

There are four basic components that are considered in VANETs

1.2.1) **Communication**:- A structure that requires a component installed within the vehicle so that it may reciprocate the information with other vehicles or it may be used as a component source for transfer of information from source to destination. In a vehicle the mode of communication requires a Wireless Access in Vehicular Environment (WAVE) which requires IEEE 1609.2 Standard also Known DSRC 802.11p [12].

1.2.2) **Multi-Hop Communication Support System**:- a multi hop communication system that is used for vehicles out of range. Here when the vehicle is within the range there is hardly any issues of data/information transfer, however incase if the range exceeds the permissible limits, hopping from one node to another is not possible. The maximum range in Dedicated Short Range Communication (DSRC) is 1000m.

1.2.3) **On-Board Unit (OBU)**:- A device which is inside the vehicle which processes the data collected from various sensors fitted inside the cars and gives conditions of the vehicles is responsible for communication with outside network i.e with other vehicles and infrastructure.

1.2.4) **Road Side Unit (RSU)**:- Infrastructure for communication between the cars for sharing and information from various vehicles. These are often considered to be the stationary points on a particular network.

1.3 Foundation of Vehicular Ad Hoc Network

The emergence of Ad Hoc Network had commenced way back in 1970. Since then there has been 4 decades passed and Vehicular Ad Hoc Network has emerged in the field of research for betterment of Globalization.

The basic history dates of these networks where they have emerged as an area of research from just a thought process are:

- **1970**: U.S. started a research project to interconnect the tactical units deployed in areas of military conflict without requiring the presence of a fixed network. The project, called PRNET (Packet Radio Network), used a combination of ALOHA and CSMA protocols, combined with a Distance Vector Algorithm.
- **1980:** Evolved into SURAN (Survivable Adaptive Radio Network), uses hierarchical routing protocol Link State Algorithm.
- **1990**: Intelligent Engineering Task Force (IETF) created MANET working group, looking for standardizing the relevant aspects of ad hoc networks to use in commercial applications.
- **2000:** An Ad Hoc Network Consortium in Japan was developed, with an aim in mind to unite the interests and efforts of industry.
- **2010:** Nowadays, it is using in many projects, especially where we cannot have a fixed infrastructure.

As we study about the infrastructure today, the first thing that comes to our mind is to know about the basic features and principles for Ad Hoc Networks. They are summarized as under:

- Dynamic topology
- Variability of the radio channel
- Do not require network infrastructure
- Using multi-hop communications
- Limited bandwidth

1.4 Vehicular Network Architectures

Vehicular network can be deployed by network operators and service providers or through integration between operators, providers, and a governmental authority. Recent advances in wireless technologies and the current and advancing trends in ad hoc network scenarios allow a number of deployment architectures for vehicular networks, in highway, rural, and city environments [16]. Such architectures should allow communication among nearby vehicles and between vehicles and nearby fixed roadside equipment. Three alternatives include (i) a pure wireless vehicle-to-vehicle ad hoc network (V2V) allowing standalone vehicular communication with no infrastructure support, (ii) a wired backbone with wireless last hops that can be seen as a WLAN-like vehicular networks, (iii) and a hybrid vehicle to-road (V2R) architecture that does not rely on a fixed infrastructure in a constant manner, but can exploit it for improved performance and service access when it is available. In this latter case, vehicles can communicate with the infrastructure either in a single hop or multi hop fashion [28] according to the vehicles' positions with respect to the point of attachment with the infrastructure. Actually the V2R architecture implicitly includes V2V communication. As a reference architecture for vehicular networks is proposed within the C2C-CC, distinguishing between three domains: in-vehicle, ad hoc, and infrastructure domain. Figure 1.1 illustrates this reference architecture. The in-vehicle domain refers to a local network inside each vehicle logically composed of two types of units: (i) an on-board unit (OBU) and (ii) one or more application unit(s) (AUs). An OBU is a device in the vehicle having communication capabilities (wireless and/or wired), while an AU is a device executing a single or a set of applications while making use of the OBU's communication capabilities [22]. Indeed, an AU can be an integrated part of a vehicle and be permanently connected to an OBU. It can also be a portable device such as a laptop or PDA that can dynamically attach to (and detach from) an OBU. The AU and OBU are usually connected with a wired connection, while wireless connection is also possible (e.g., Bluetooth, WUSB, or UWB). This distinction between AU and OBU is logical, and they can also reside in a single physical unit. The ad hoc domain is a network composed of vehicles equipped with OBUs and road side units (RSUs) that are stationary along the road. OBUs of different vehicles form a mobile ad hoc network (MANET), where an OBU is equipped with communication devices, including at least a short range wireless communication device dedicated for road safety. OBUs and RSUs can be seen as nodes of an ad hoc network, respectively, mobile and static nodes. An RSU can be attached to an infrastructure network, which in turn can be connected to the Internet. RSUs can also communicate to each other directly or via multi hop, and their primary role is the improvement of road safety, by executing special applications and by sending, receiving, or forwarding data in the ad hoc domain. Two types of infrastructure domain access exist: RSU and hot spot. RSUs may allow OBUs to access the infrastructure, and consequently to be connected to the Internet. OBUs may also communicate with Internet via public, commercial, or private hot spots (Wi-Fi hot spots) [33]. In the absence of RSUs and hot spots, OBUs can utilize communication capabilities of cellular radio networks (GSM, GPRS, UMTS, Wi-Max and 4G) if they are integrated in the OBU.

1.5 Characteristics of Vehicular Ad Hoc Networks

Vehicular networks have special behavior and characteristics, distinguishing them from other types of mobile networks. In comparison to other communication networks, vehicular networks come with unique attractive features, as follows:

1.5.1) **Unlimited transmission power** :- Mobile device power issues are usually not a significant constraint in vehicular networks as in the case of classical ad hoc or sensor networks, since the node (vehicle) itself can provide continuous power to computing and communication devices.

1.5.2) **Higher computational capability** : - Indeed, operating vehicles can afford significant computing, communication, and sensing capabilities. Predictable mobility: Unlike classic mobile ad hoc networks, where it is hard to predict the nodes' mobility, vehicles tend to have very predictable movements that are (usually) limited to roadways. Roadway information is often available from positioning systems and map based technologies such as GPS. Given the average speed, current speed, and road trajectory, the future position of a vehicle can be predicted.

1.6 Challenging Characteristics

However, to bring its potency to fruition, vehicular networks have to cope with some challenging characteristics, which include:

1.6.1) **Potentially large scale** :- Unlike most ad hoc networks studied in the literature that usually assume a limited network size, vehicular networks can in principle extend over the entire road network and so include many participants.

1.6.2) **High mobility** :- The environment in which vehicular networks operate is extremely dynamic, and includes extreme configurations: on highways, relative speeds of up to 300 km/h may occur, while density of nodes may be 1–2 vehicles 1 km on low busy roads [19]. On the other hand, in the city, relative speeds can reach up to 60 km/h and nodes' density can be very high, especially during rush hour.

1.6.3) **Partitioned Network** :- Vehicular networks will be frequently partitioned. The dynamic nature of traffic may result in large inter vehicle gaps in sparsely populated scenarios, and hence in several isolated clusters of nodes.

1.6.4) **Network Topology and Connectivity**: - Vehicular network scenarios are very different from classic ad hoc networks. Since vehicles are moving and changing their position constantly, scenarios are very dynamic. Therefore the network topology changes frequently as the links between nodes connect and disconnect very often. Indeed, the degree to which the network is connected is highly dependent on two factors: the range of wireless links and the fraction of participant vehicles, where only a fraction of vehicles on the road could be equipped with wireless interfaces.

1.7 Vehicular Network Potential Applications and Services

Vehicular network applications range from road safety applications oriented to the vehicle or to the driver, to entertainment and commercial applications for passengers, making use of a plethora of cooperating technologies. The primary vision of vehicular networks includes real-time and safety applications for drivers and passengers, providing safety for the latter and giving essential tools to decide the best path along the way [7]. These applications thus aim to minimize accidents and improve traffic conditions by providing drivers and passengers with useful information including collision warnings, road sign alarms, and in-place traffic view.

Nowadays, vehicular networks are promising in a number of useful driver- and passenger-oriented services, which include Internet connections facility exploiting an available infrastructure in an “on-demand” fashion, electronic tolling system, and a variety of multimedia services. As well as, a variety of communication networks, such as 2-3G, WLANs IEEE 802.11a/b/g/p and Wi-MAX can be exploited to enable new services designed for passengers apart from the safety applications, such as info mobility and entertainment applications, which can rely on the vehicular network itself [34]. Regarding the discussed applications’ potential, vehicular networks open new business opportunities for car manufacturers, automotive OEMs, network operators, service providers, and integrated operators in terms of infrastructure deployment as well as service provision and commercialization. For safety-related applications, the network operator can assure the authentication of each participant through playing the role of a trusted third party that authenticates the participating nodes, or even having the role of a certification authority issuing a certificate to each participant in order to prove the authenticity of them later during the communication. On the other hand, in non safety-related applications, network operators and/or service providers, besides network access and services’ provision can have the role of authorizing services’ access and billing users for the consumed services [2]. However, one should notice that ad hoc systems still require a certain level of penetration and necessitate high vehicle density for more reliable communication. Also, the investment cost for new communication infrastructure for vehicular networks is high, whereas on the other hand cellular communication systems offer a high coverage along the roads and have a reliable authentication and security mechanism. Consequently number of technical challenges needs to be resolved in order to help the evolution of vehicular networks for wide-scale deployment. The following section discusses some of these challenges.

1.8 Advantages and Disadvantages of Ad Hoc Networks

Since Ad Hoc Networks has really enhanced the experience of transfer of data from one point to another point, there are also some limitations in its consideration. Following considerations have been identified on individual basis.

Advantages	**Disadvantages**
Independence from central network administration.	Each node must have full performance.
Self-configuring, nodes are also routers.	Throughput is affected by system loading.
Self-healing through continuous re-configuration.	Reliability requires a sufficient number of available nodes. Sparse networks can have problems.
Scalable: accommodates the addition of more nodes.	Large networks can have excessive latency (high delay), which affects some applications access the Internet from many different locations.
Flexible: similar to being able to access the Internet from many different locations.	Information hopping may delay the reaching time from source to destination.

Table 1.1 Advantages and disadvantages of Ad Hoc Network

Conclusion

In this chapter we have understood the basic fundamentals of all the technical terminology of the key aspects that we are going to make use of or experiment upon. The basic need is to be sure about the self guiding drawbacks from the existing system, from where the scope for further improvement enhances, more over the same aspect could be found in the recent researches available in today's date on the subject scenario. Hence in the next chapter we would study literature and experiments incorporated on network ad hoc systems and would try to identify a feasible possibility by matching the common aspects between the drawbacks and shortcomings in the till date by introducing a new system out of the existing system.

Chapter 2

Literature Survey

Introduction

In this chapter, we have carried out detailed study based on the achievements incorporated on Trajectory based Data Forwarding and now our taxonomy is to understand the step by step technique of its functioning by guiding our self by a defined literature survey. The most important thing in the entire survey is to identify that the use of nearest node has been taken in consideration for passing of information and no standard prescribed tests have been adopted for the existing system. Once the study of the existing system is over we would identify the problem from the driven solution and would use it as a foundation for future design and an improvement over the existing system.

2.1 Taxonomy of Trajectory based Data Forwarding

Vehicular Ad Hoc Networks (VANETs) have recently reemerged as one of promising research areas for safety and connectivity in road networks. Currently, most research and development fall into one of two categories: (i) vehicle-to-vehicle (v2v) communications and (ii) vehicle-to-infrastructure (v2i) communications. In the meantime, the GPS technology has been adopted for navigation purposes at an unprecedented rate [5]. Specifically, this work is motivated by the observed trend that a large number of vehicles have started to install GPS receivers for navigation and the drivers are guided by these GPS-based navigation systems to select better driving paths in terms of the physically shortest path or the vehicular low density traffic path. Therefore, the nature research question is how to make the most of this trend to improve the performance of vehicular ad hoc networks. The first challenge is how to use the trajectory information in a privacy-preserving manner, while improving the data forwarding performance. To resolve this challenge, we design a local algorithm to compute Expected Data Delivery Delay (EDDD) at individual vehicles to an access point, using private trajectory information and known traffic statistics. Only the computed delay is shared with neighboring vehicles. The vehicle with the shortest EDDD is selected as the next packet carrier for its neighboring vehicles. The other challenge is how to model an accurate road link delay, a delay defined as the time taken for a packet to travel through a road segment using carry-and-forward [31]. To resolve this challenge, we accurately model road link delay, based on traffic density information obtained from the GPS-based navigation system. Our intellectual contributions are as follows:

• An analytical link delay model for packet delivery along a road segment that is much more accurate than that of the state-of-the-art solution.
• An expected E2E delivery delay computation based on individual vehicle trajectory for the better decision making on the packet forwarding.

2.2 Trajectory based Data Forwarding Environment

Given a road network with an Internet access point, the research problem is to minimize the end-to-end delivery delay of packets to the Internet access point. In this paper, we focus on one-way data delivery which is useful for the time-critical reports, such as vehicle accidents, road surface monitoring and driving hazards. We leave two-way delivery as future work. In this paper, we refer Vehicle trajectory as the moving path from the vehicle's starting position [3] to its destination position in a road network; (ii) Expected Delivery Delay (EDD) as the expected time taken to deliver a packet generated by a vehicle to an Internet access point via the VANET; (iii) Carry delay as a part of the delivery delay introduced while a packet is carried by a moving vehicle; (iv) Communication delay as a part of the delivery delay introduced while a packet is forwarded among vehicles. Our work is based on the following four assumptions:

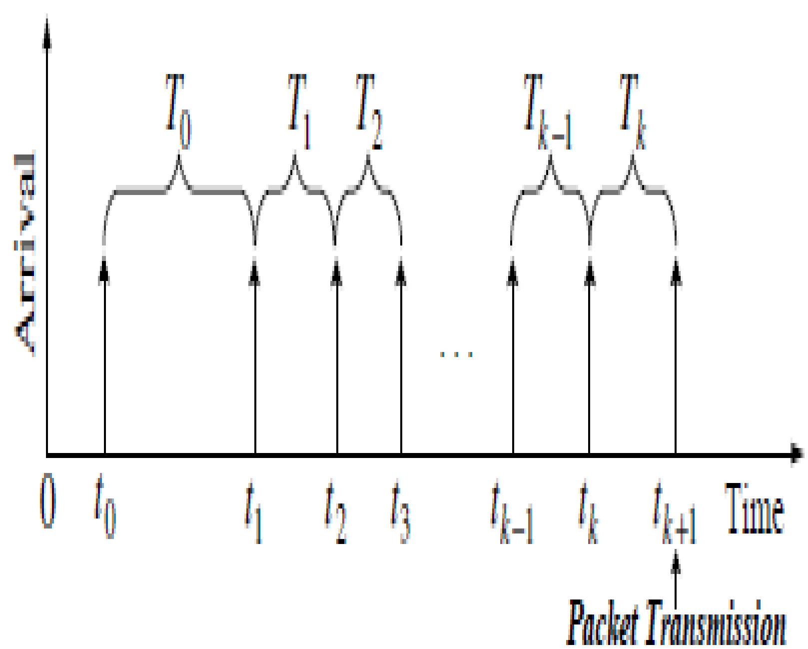
• The geographical location information of packet destinations, such as Internet access points (APs), is available to vehicles. A couple of studies have been done to utilize the Internet access points available on the road-sides.

• Vehicles participating in VANET have a wireless communication device, such as the

Dedicated Short Range Communications (DSRC) device. Nowadays many vehicle vendors, such as GM and Toyota, are planning to install DSRC devices at vehicles.

• Vehicles are installed with a GPS-based navigation system and digital road maps. Traffic statistics, such as vehicle arrival rate and average vehicle speed v per road segment, are available via a commercial navigation service, such as Garmin Traffic.

• Vehicles know their trajectory by themselves. However, vehicles do not release their trajectory to other vehicles for privacy concerns. It should be noted that in the VANET scenarios, the carry delay is several orders-of-magnitude longer than the communication delay. For example, a vehicle takes 90 seconds to travel along a road segment of 1 mile with a speed of 40 MPH, however, it takes only ten of milliseconds 1 to forward a packet over the same road segment, even after considering the retransmission due to wireless link noise or packet collision [6]. Therefore, since the carry delay is the dominating part of the total delivery delay, in the rest of the paper we focus on the carry delay for the sake of clarity, although the small communication delay does exist in our design. In the next sections, we will explain our Link delay modeling and Trajectory-based data forwarding.



(b) Vehicle Arrival Sequence on One-way Road Segment

Figure 2.1 Vehicle arrival sequence on one way Road Segment

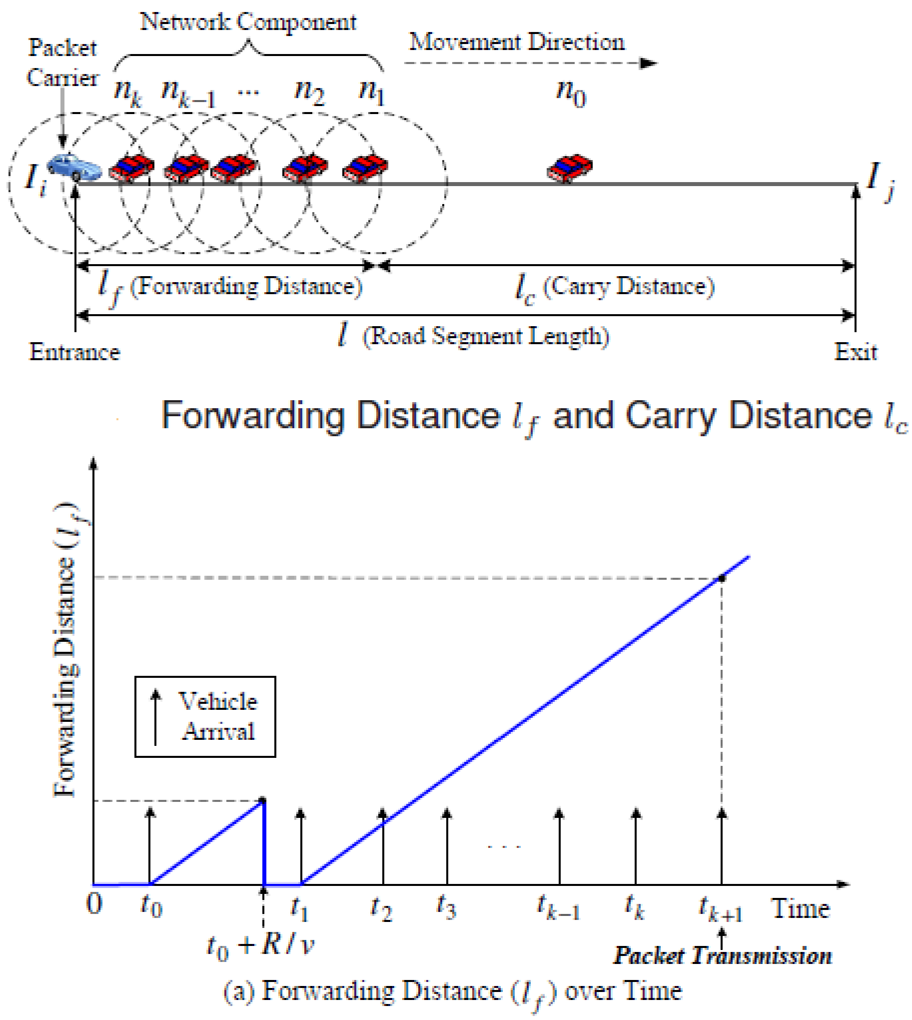


Figure 2.2 Trajectory Based Data Forwarding

2.3 End to End Delay

We model the EDD with a stochastic model for a given road network. We define the road network graph for the EDD computation as follows:

Road Network Graph Let a road network graph be the directed graph of G = (V,E), where V = {v1, v2, ..., vn} is a set of intersections in the road network and E = [eij] is a matrix of edge eij for vertices vi and vj such that eij 6= eji. Following figure …..shows a road network graph.

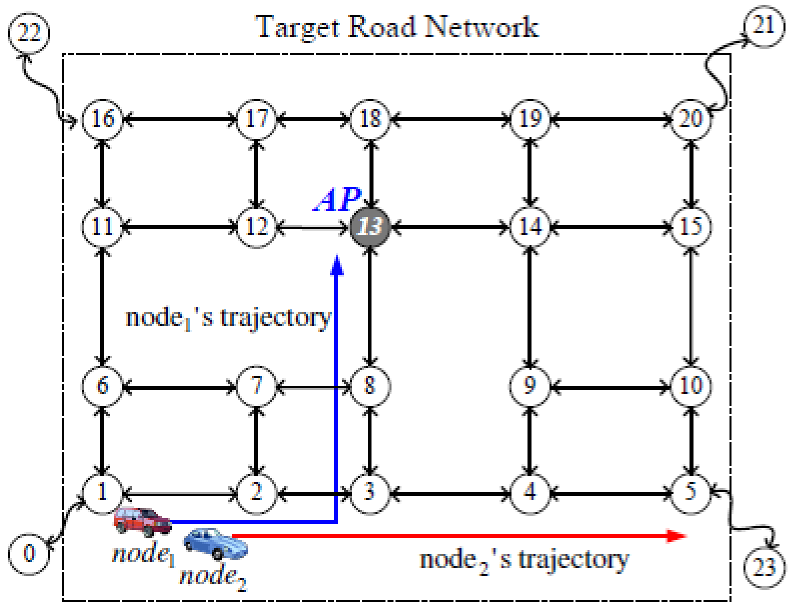


Figure 2.3 Network Map of an area

To estimate end-to-end delay, we cannot use the traditional shortest path algorithms, such as Dijkstra's shortest path algorithm. This is because when the packet carrier arrives at an intersection, it is not guaranteed that it can meet another vehicle moving towards the most preferred direction. In this case, the packet carrier needs to determine whether it can forward its packet to another vehicle moving towards other preferred directions or has to carry it with itself to the next intersection on its trajectory. In order to consider all of the possible cases in the forwarding at each intersection [14], we formulate the data delivery based on this carry-and-forward as the stochastic model.

2.4 Expected Delivery Delay at Intersection

In this section, we explain how to compute the EDD at an intersection, using a stochastic model. Suppose that a packet at intersection is delivered towards intersection j. Let dij be the link delay for edge eij in Equation 1. We note the expected delay EDD at an intersection depends on the forwarding direction (i.e., edge). Therefore, we use Dij denote the EDD at the intersection i when the edge eij is used as the forwarding edge. We formulate Dij recursively as follows:

$$D_{ij} = d_{ij} + E \text{ [delivery delay at j by forwarding or carry]}$$
$$= d_{ij} + \sum_{k\epsilon n(J)} (Pjk\ Djk)$$

where N(j) is the set of neighboring intersections of intersection j. We use this stochastic model to compute the EDD at intersection i because the packet will be delivered with some probability to one of outgoing edges at intersection j. This means that when the carrier of this packet arrives at intersection j, the next carrier on each outgoing edge towards intersection k will be met with probability Pjk. We will explain how to compute the probability Pjk later.

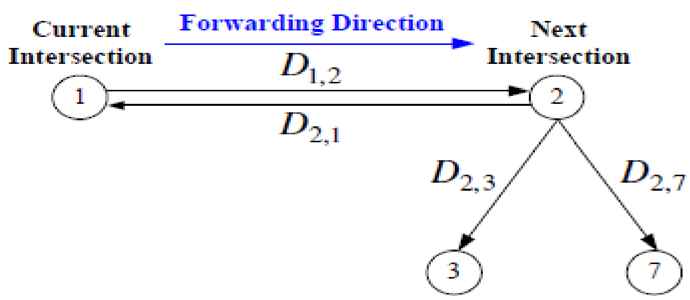


Figure 2.4 Vehicle's direction towards intersection

Here is the situation where the problem lies. We will be finding out the way outs to rectify the problem and to improve the current situation.

Conclusion

Since we have studied the existing study on the basis of TBDD, we are able understand the basic fundamental for the reason of adopting this kind of a procedure for work where the data transfer relies on hopping onto a node going in the similar direction of the need based data transfer. However, the information gets transferred from one node to another depending upon the expected delays of the node, or due to surpassing of the defined path. There is an utter need of developing a function based activity that would make use of the current functionaries installed in the nodes and then transfer the information to a node.

Chapter 3

Problem Statement

Introduction

In previous chapter, we had shed light on the various updates available in today's date for trajectory based data forwarding technique and also identified some of the lags in the existing study. Hence keeping in view to the current requirement of the hour, we would identify the specific reasons for the data latency during information transfer and would discuss the best possible solution to the current problem.

3.1 Gap Analysis

In the current environment, we have learnt that Trajectory based Data Forwarding has been successful so far however there are some constraints with regard to the nodes that travel on an available network, the problem related to the existing working comes becomes a constraint when a node instead of taking an optimal path, selects the path ambiguously. The objective is achieved, however there are not potential benefits extracted from the available solution. Hence when we calculate the entire process in terms of a satisfactory evaluation, the entire process is not fruitful. The concerned reason behind it is that the node selects the route as to destination without any evaluation of data. Moreover on the very node we are also transferring data signals [20]. Hence as a result the there exists a latency of delivery of packets. In the same regard the future process that is involved to the operation, also gets delayed. Hence, after evaluating the existing process following drawbacks have been analyzed.

a) Results are satisfactory, however if it the entire process is done on the basis of evaluation, we can achieve a faster navigation of data from one part of the network to another part (source to destination).

b) Analytical link delay on the information among the network model for packet delivery not available.

c) Expected End-to-End (E2E) delivery delay based on individual node, instead of number of nodes available within one network.

With the above said facts, there are also some of the short comings which are not ignorable, hence we can calculate the following shortcomings in following sum points.

a) Delay in delivery time occurs, as existing information is not evaluated prior to the operation of data hop.

b) Wrong path adopted to travel within the network.

c) Moving from a path that does not have a relay/access which provides post operational support.

d) Overall communication delay of data due to various operational situations that are encountered while the node travels within the network.

3.2 Objectives of Dissertation

As it is often quoted that, " There is always scope for improvement ", we have deeply analyzed the existing problem and tried to improve on existing system by keeping emphasize of all the resources that are available within the existing system, however are overlooked or ignored. The proposed functioning goes as follows:

- To propose a system that provides navigation from source to destination choosing the **Shortest Path** available, with minimum data hop, by calculating values at all paths, and then select a path to travel which provides reasonably improved outcome.

- To use the **GPS System** (already installed in nodes[vehicles]) to acquire the data that the driver has embedded to select the path to travel and on the basis of evaluation of the information for route, may determine whether sharing of information that needs to be transferred will be feasible or not.

- To ensure that there is no **Data Latency** while the node is traveling from source to origin.

- To undertake major issues of **Data Accuracy** and **Data Delivery Speed** once the data is transferred at destination address.

- To ensure the D**elivery Ratio** attains success by accurate delivery.

As seen in existing study the fundamental that uses node as a medium of object to transfer the information from one part of the network to another of the existing system gives a unique model of ideology that there is a lot of scope of improvement to the existing system. Hence keeping all the aspects of the existing system and requirement for up gradation that may lead to a newer interface we have come up with a new technique referred to as, "Trajectory Controlled Data Dissemination for VANETs." In addition to the algorithm we have also made use of the Dijgastra's algorithm.

Dijkstra's algorithm was conceived by computer scientist Edsger Dijkstra in 1956 and published in 1959. It is a graph search algorithm that solves the single-source shortest path problem for a graph with non-negative edge path costs, producing a shortest path tree. This algorithm is often used in routing and as a subroutine in other graph algorithms.

For a given source vertex (node) in the graph, the algorithm finds the path with lowest cost (i.e. the shortest path) between that vertex and every other vertex. It can also be used for finding costs of shortest paths from a single vertex to a single destination vertex by stopping the algorithm once the shortest path to the destination vertex has been determined. For example, if the vertices of the graph represent cities and edge path costs represent driving distances between pairs of cities connected by a direct road, Dijkstra's algorithm can be used to find the shortest route between one city and all other cities. As a result, the shortest path algorithm[22] is widely used in network routing protocols, most notably IS-IS and OSPF (Open Shortest Path First).

Dijkstra's original algorithm does not use a min-priority queue and runs in time O(|V|^2) (where |V| is the number of vertices). The implementation based on a min-priority queue. This is asymptotically the fastest known single-source shortest-path algorithm for arbitrary directed graphs with unbounded non-negative weights.

Dijkstra's algorithm search for finding path from a start node to a goal node in a robot motion planning problem. Open nodes represent the "tentative" set. Filled nodes are visited ones, with color representing the distance: the greener, the farther. Nodes in all the different directions are explored uniformly appearing as a more-or-less circular wave front as Dijkstra's algorithm uses a heuristic identically equal to 0.

The basic fundamental for the algorithm is that the node at which we are starting be called the Initial node. Let the distance of node Y is the distance from the initial node to Y. Dijkstra's algorithm will assign some initial distance values and will try to improve them step by step.

Assign to every node a tentative distance value: set it to zero for our initial node and to infinity for all other nodes. Mark all nodes unvisited. Set the initial node as current. Create a set of the unvisited nodes called the unvisited set consisting of all the nodes.

For the current node, consider all of its unvisited neighbors and calculate their tentative distances. Compare the newly calculated tentative distance to the current assigned value and assign the smaller one. For example, if the current node A is marked with a distance of 6, and the edge connecting it with a neighbor B has length 2, then the distance to B (through A) will be 6 + 2 = 8. If B was previously marked with a distance greater than 8 then change it to 8. Otherwise, keep the current value observing till the post value occurs.

When we are done considering all of the neighbors of the current node, mark the current node as visited and remove it from the unvisited set. A visited node will never be checked again.

If the destination node has been marked visited (when planning a route between two specific nodes) or if the smallest tentative distance among the nodes in the unvisited set is infinity (when planning a complete traversal; occurs when there is no connection between the initial node and remaining unvisited nodes), then stop. The algorithm has finished.

Select the unvisited node that is marked with the smallest tentative distance, and set it as the new "current node" then go back to previous step.
The easier description for the operational module in a comfortable manner we had made use of the terms intersection, road and map — however, in formal notation these terms are vertex, edge and graph, respectively.

If we would like to find the shortest path between two intersections on a city map, a starting point and a destination. The order is conceptually simple: to start, mark the distance to every intersection on the map with infinity. This is done not to imply there is an infinite distance, but to note that intersection has not yet been visited; some variants of this method simply leave the intersection unlabeled. Now, at each iteration, select a current intersection. For the first iteration, the current intersection will be the starting point and the distance to it (the intersection's label) will be zero. For subsequent iterations (after the first), the current intersection will be the closest unvisited intersection to the starting point—which would be easy to find.

From the current intersection, update the distance to every unvisited intersection that is directly connected to it. This is done by determining the sum of the distance between an unvisited intersection and the value of the current intersection, and relabeling the unvisited intersection with this value if it is less than its current value. In effect, the intersection is relabeled if the path to it through the current intersection is shorter than the previously known paths. To facilitate shortest path identification, in pencil, mark the road with an arrow pointing to the relabeled intersection if you label/relabeled it, and erase all others pointing to it. After you have updated the distances to each neighboring intersection, mark the current intersection as visited and select the unvisited intersection with lowest distance (from the starting point) – or the lowest label—as the current intersection. Nodes marked as visited are labeled with the shortest path from the starting point to it and will not be revisited or returned to.

Continue this process of updating the neighboring intersections with the shortest distances, then marking the current intersection as visited and moving onto the closest unvisited intersection until you have marked the destination as visited. Once you have marked the destination as visited (as is the case with any visited intersection) you have determined the shortest path to it, from the starting point, and can trace your way back, following the arrows in reverse.

Of note is the fact that this algorithm makes no attempt to direct "exploration" towards the destination as one might expect. Rather, the sole consideration in determining the next "current" intersection is its distance from the starting point. This algorithm, therefore "expands outward" from the starting point, interactively considering every node that is closer in terms of shortest path distance until it reaches the destination. When understood in this way, it is clear how the algorithm necessarily finds the shortest path, however, it may also reveal one of the algorithm's weaknesses: its relative slowness in some topologies.

Conclusion

Moving from the existing system to propose system is the basic fundamental of improvement required. Hence keeping the existing problem in mind we have identified the issue related to the VANETs transmission and have also introduced a two phase solution for the entire system. The first phase of remedy has been vividly discussed in the system, by giving a liberty to the existing system by identifying the shortest distance that needs to be traveled by a node, in an order to reach from host to destination based on the weights at different path. Achieving the above said objective would almost diminish the Data Latency issue by 50% of the overall delay.

Chapter 4

Trajectory Controlled Data Dissemination Algorithm for VANETs

Introduction

In this phase we would discuss the second phase of improvement by proposing an algorithm on the existing system so as to achieve the basic target of removing the sensitive issues of data latency and accuracy of the data delivery by introducing, “Trajectory Controlled Data Dissemination (TCDD) algorithm for VANETs.”

4.1 Trajectory Controlled Data Dissemination (TCDD) algorithm for VANETs

Trajectory Controlled Data Dissemination (TCDD) algorithm for VANETs has been embedded keeping the fundaments of the existing system, where new aspects on available system have been applied by intervening with the interface of node. TCDD further advances to newer heights, as it takes in consideration a two step methodology. The first methodology is based on evaluating the path by calculating weights at each node, within the connecting network. The second step theory in the algorithm is to incorporate with the GPS system in the vehicle and identify the route that is being taken by the vehicle to reach to the destination. If the route of the vehicle matches with the weight calculated on each path, the information that needs to be transferred will be shifting the base from the relay station to the vehicle and when the vehicle reaches or crosses the destination of the informational node, the information is de-boarded from the vehicle and hence the prime object of creating the entire operation of sending the information from one part of the network to another is achieved. To understand the same let us explain in a diagrammatic manner.

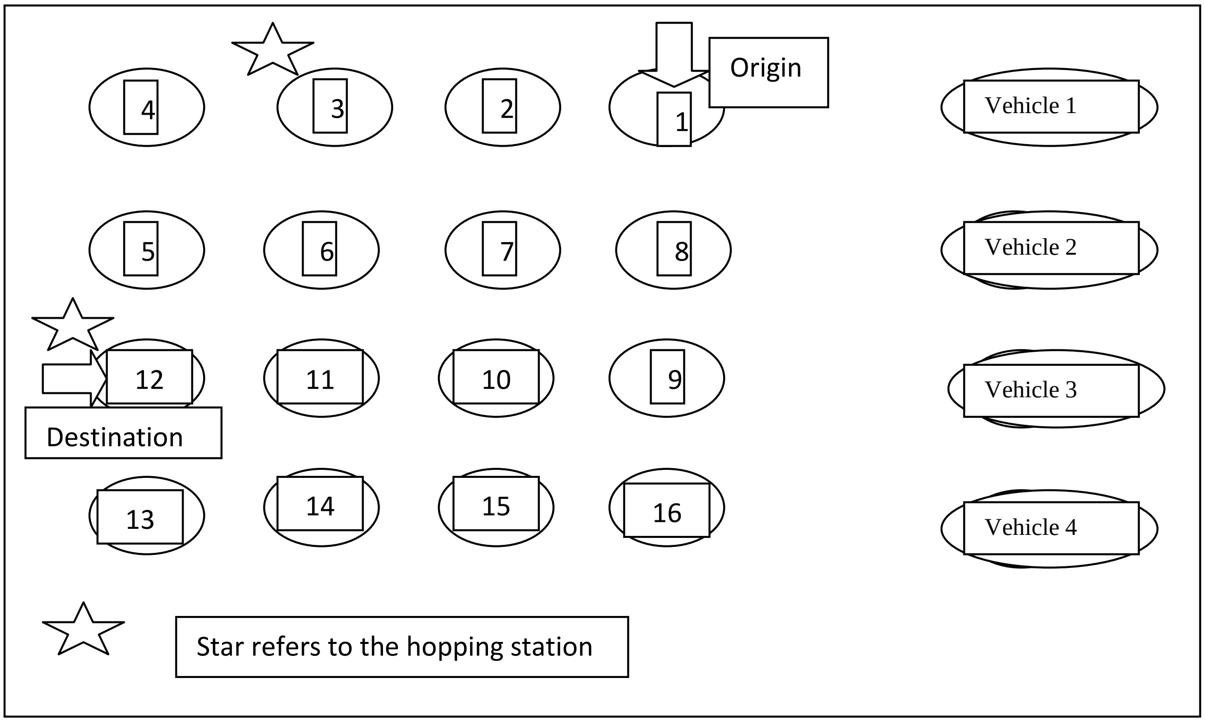


Figure 4.1: Network Area with different nodes

Here in Figure 4.1 the area shows the different paths available for vehicles to travel in an area. Here there are two hopping points (stations), one at 3 and another at 12. They are made to avoid information boarded on a node to be transmitted on a forbidden path i.e. incase if the vehicle goes astray than the dedicated path from source to origin, the information from the vehicle will be hopped on the relay station and will be stationary until another node is selected and is evaluated on the basis of pre installed GPS system that helps the node in navigation of the node. Hence when the vehicle passes from the relay station the built in formulation firstly evaluates the node's destination, and then evaluates the path that the node has selected to reach to the destination. If the destination matches with the requirement, the second stage is to evaluate the shortest path selected by node, if the node has not selected the path that is not satisfying the path, then the information from the relay station is not hopped onto the node, and waits for another node to encounter and evaluate the node in the same manner, in which the pervious node was evaluated and incase if the parameters are satisfied, the information is sent over onto the node.

The TCDD algorithm has various steps of formation in the entire course of operation. Let us first understand in technical terms.

Evaluation Steps

Step 1. Consider a Network Area (N/W) having various vehicles (V) that travel within the network.

Step 2. Calculate the weight at paths at each vertex (VE) within the N/W.

Step 3. Save the weights from vertex to vertex in an Edge Table for faster reference.

Step 4. Consider a packet (P) being stored at VE on path location number 3.

Step 5. Consider vehicle (VEH) a medium traveling in N/W.

Step 6. When VEH enters the N/W the network mechanism intervenes with the GPS system of the vehicle that is passing within the N/W.

Step 7. If VEH has selected the same path where the information needs to be transferred, the VEH evaluation enters the second phase of evaluation for selection, where the exact path selected by the vehicle is evaluated on the basis of Shortest Path.

Step 8. If the vehicle fairs in the evaluation, the information that needs to be transferred is hopped from the resource location to vehicle.

Step 9. If the vehicle as selected a longer path for traveling from source to destination, then the relay device waits for another vehicle to come to the origin and on vehicle steps 6 to 8 are evaluated.

In the following algorithm, the code u := vertex in Q with min dist[u], searches for the vertex u in the vertex set Q that has the least dist[u] value. length(u, v) returns the length of the edge joining (i.e. the distance between) the two neighbor-nodes u and v. The variable alt on line 17 is the length of the path from the root node to the neighbor node v if it were to go through u. If this path is shorter than the current shortest path recorded for v, that current path is replaced with this alt path. The previous array is populated with a pointer to the "next-hop" node on the source graph to get the shortest route to the source.

Algorithm

```
function TCDD(Graph, source):
dist[source]  := 0                    // Distance from source to source
 for each vertex v in Graph:           // Initializations
if v ? source
dist[v]    :=  infinity                 //  Unknown  distance  function  from  source  to  v  6
previous[v]  := undefined      // Previous node in optimal path from source
end if
 add v to Q                        // All nodes initially in Q
 end for
while Q is not empty:                  // The main loop
u := vertex in Q with min dist[u]  // Source node in first case
remove u from Q
for each neighbor v of u:           // where v has not yet been removed from Q.
 alt := dist[u] + length(u, v)
if alt < dist[v]:               // A shorter path to v has been found
dist[v]  := alt
previous[v]  := u
end if
end for
end while
return dist, previous
end function
```

If we are only interested in a shortest path between vertices source and target, we can terminate the search at line 13 if u = target. Now we can read the shortest path from source to target by reverse iteration:

```
S := empty sequence
u := target
while previous[u] is defined:              // Construct the shortest path with a stack S
 insert u at the beginning of S          // Push the vertex into the stack
 u := previous[u]                        // Traverse from target to source
end while
```

Now sequence S is the list of vertices constituting one of the shortest paths from source to target, or the empty sequence if no path exists.

A more general problem would be to find all the shortest paths between source and target (there might be several different ones of the same length). Then instead of storing only a single node in each entry of previous we would store all nodes [25] satisfying the relaxation condition. For example, if both r and source connect to target and both of them lie on different shortest paths through target (because the edge cost is the same in both cases), then we would add both r and source to previous [target]. When the algorithm completes, previous data structure will actually describe a graph that is a subset of the original graph with some edges removed. Its key property will be that if the algorithm was run with some starting node, then every path from that node to any other node in the new graph will be the shortest path between those nodes in the original graph, and all paths of that length from the original graph will be present in the new graph. Then to actually find all these shortest paths between two given nodes we would use a path finding algorithm on the new graph, such as depth-first search.

Using a priority queue

A min-priority queue is an abstract data structure that provides 3 basic operations : add_with_priority(), decrease_priority() and extract_min(). As mentioned earlier, using such a data structure can lead to faster computing time than using a basic queue. Notably

Fibonacci heap (Fredman & Tarjan 1984) or Brodal queue offer optimal implementations for those 3 operations. As the algorithm is slightly different, we mention it here, in pseudo-code as well :

It should be noted that other data structures can be used to achieve even faster computing times in practice.

4.2 Running Time

An upper bound of the running time of Dijkstra's algorithm on a graph with edges E and vertices V can be expressed as a function of |E| and |V| using big-O notation.
For any implementation of vertex set Q the running time is in $O(|E| \cdot dk_Q + |V| \cdot em_Q)$, where dk_Q and em_Q are times needed to perform decrease key and extract minimum operations in set Q, respectively.

The simplest implementation of the Dijkstra's algorithm stores vertices of set Q in an ordinary linked list or array, and extract minimum from Q [29] is simply a linear search through all vertices in Q. In this case, the running time is $O(|E| + |V|^2) = O(|V|^2)$.
For sparse graphs, that is, graphs with far fewer than $O(|V|^2)$ edges, Dijkstra's algorithm can be implemented more efficiently by storing the graph in the form of adjacency lists and using a self-balancing binary search tree, binary heap, pairing heap, or Fibonacci heap as a priority queue to implement extracting minimum efficiently. With a self-balancing binary search tree or binary heap, the algorithm requires $\Theta((|E| + |V|) \log |V|)$ time (which is dominated by $\Theta(| E | \log | V |)$, assuming the graph is connected). To avoid $O(|V|)$ look-up in decrease-key step on a vanilla binary heap [10], it is necessary to maintain a supplementary index mapping each vertex to the heap's index (and keep it up to date as priority queue Q changes), making it take only $O(\log|V|)$ time instead. The Fibonacci heap improves this to $O(|E| + |V| \log|V|)$.

Note that for directed acyclic graphs, it is possible to find shortest paths from a given starting vertex in linear time, by processing the vertices in a topological order, and calculating the path length for each vertex to be the minimum length obtained via any of its incoming edges.

4.3 Related Problems and Algorithms

The functionality of Dijkstra's original algorithm can be extended with a variety of modifications. For example, sometimes it is desirable to present solutions which are less than mathematically optimal. To obtain a ranked list of less-than-optimal solutions, the optimal solution is first calculated. A single edge appearing in the optimal solution is removed from the graph, and the optimum solution to this new graph is calculated. Each edge of the original solution is suppressed in turn and a new shortest-path calculated [11]. The secondary solutions are then ranked and presented after the first optimal solution. Dijkstra's algorithm is usually the working principle behind link-state routing protocols, OSPF and IS-IS being the most common ones.

Hence the process steps are calculated on the basis of these three phase evaluation where the data is being transferred. The entire process may be easy when represented diagrammatically. Let us take one figure at a time to understand the concept more conceptually.

Here when we observe a figure we may see that there exist a path from one node to another or a node may indirectly be connected to another node or to visit to one node may need an intermediate node and indeed which is indeed to have a stronger kind of a network as more the nodes within a network more is the probability of having scattered area of network.

Hence when we observe the diagrammatic figures for each of the network, we may find that the selection of a path from one network to another may be accurate on part of individual vehicle's calculation, however that does not matches with the calculated shortest path value as per the evaluation of data transfer selection.

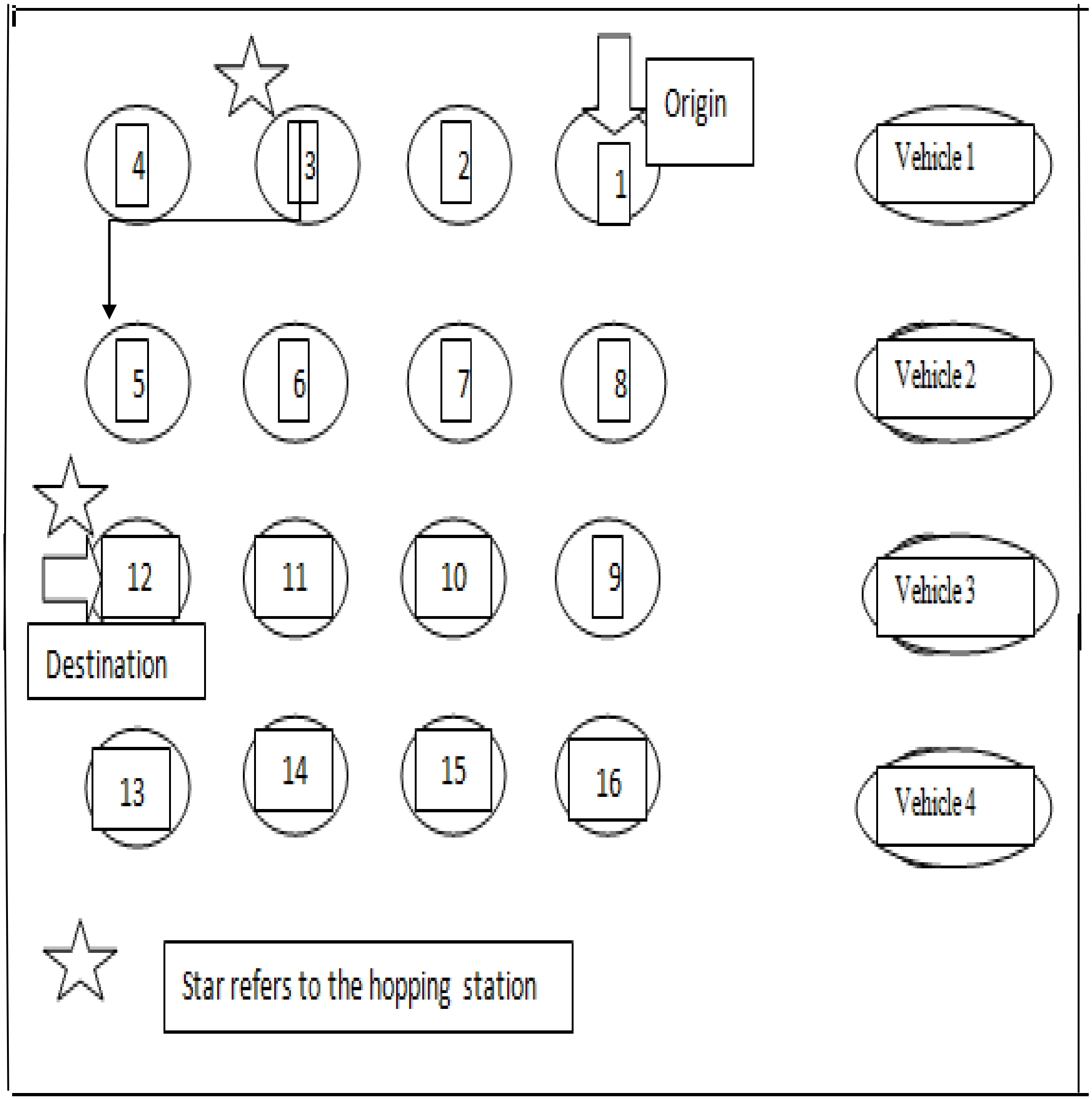


Figure 4.2 Path selected for transfer of information on basis of Shortest Path Algorithm

Here in figure 4.2 there is information stored at hopping station at vertex 3 and vertex 12. As per the nature of programming technique the program evaluates weights of the entire vertex that are within the network area that lead to destination vertex. Here any vehicle that passes through the stationary vertex is evaluated on the basis of pre-determined path selected by the vehicle after viewing the source of GPS system. A node whose path value is equal to the vertex values selected by the determined path of the information transfer, the node is taken into consideration, else the node is dropped, and another incoming node is evaluated and until then the data is kept intact with the hopping station.

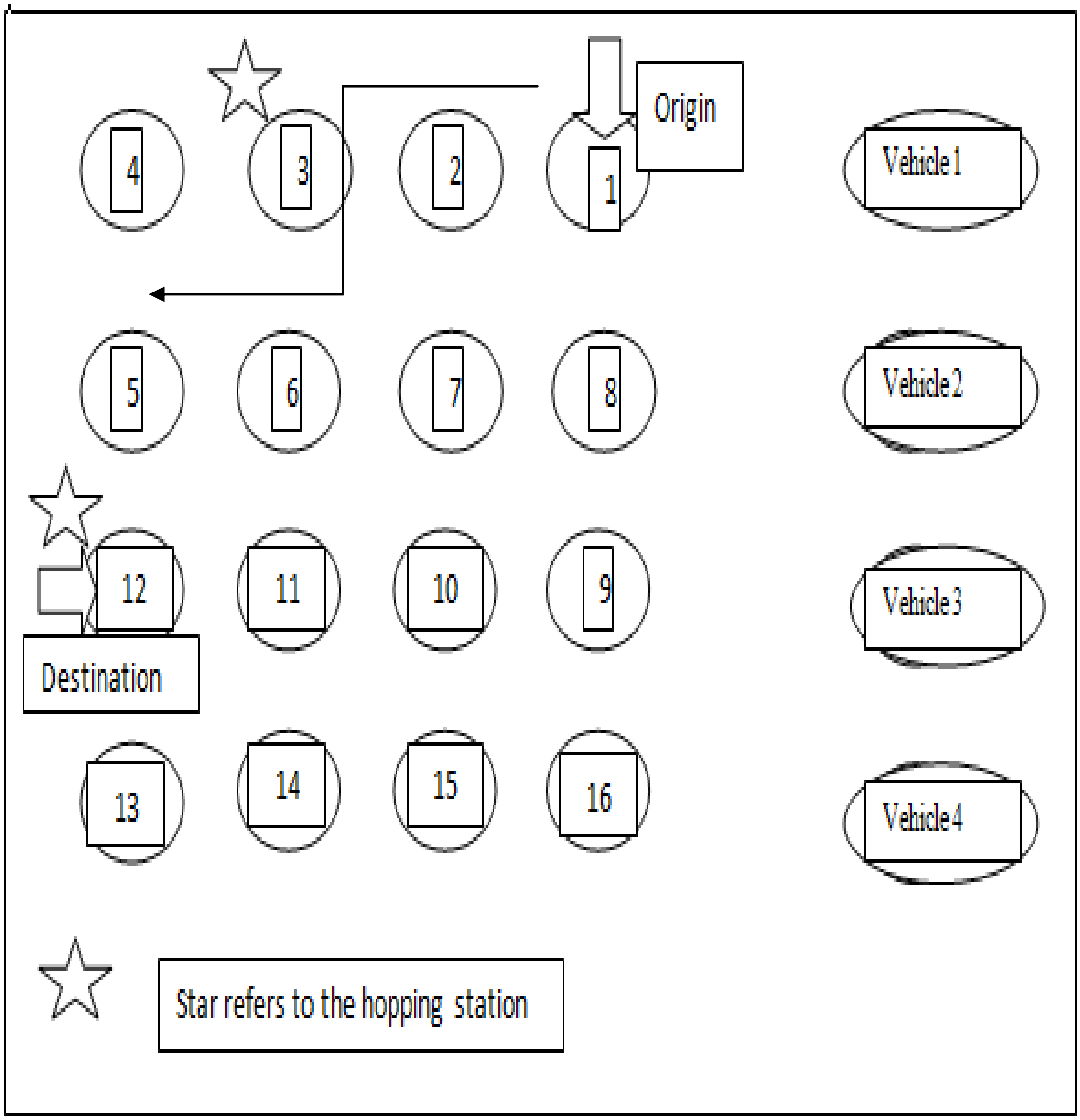


Figure 4.3 Path selected by Node (Vehicle 1) to travel from source to destination

Here in the figure 4.3 Vehicle 1 is traveling from the origin to destination and the vehicle surpasses the vertex 3 which is bearing some information that needs to be transmitted. When the vehicle reaches the vertex 3, the inbuilt GPS system is evaluated by the node, and the path that is selected by the vehicle is taken into consideration.

Since the path selected by the vehicle does not match with the path that has been selected by the End to End delivery vertex the information is not hopped and wait for next vehicle is done.

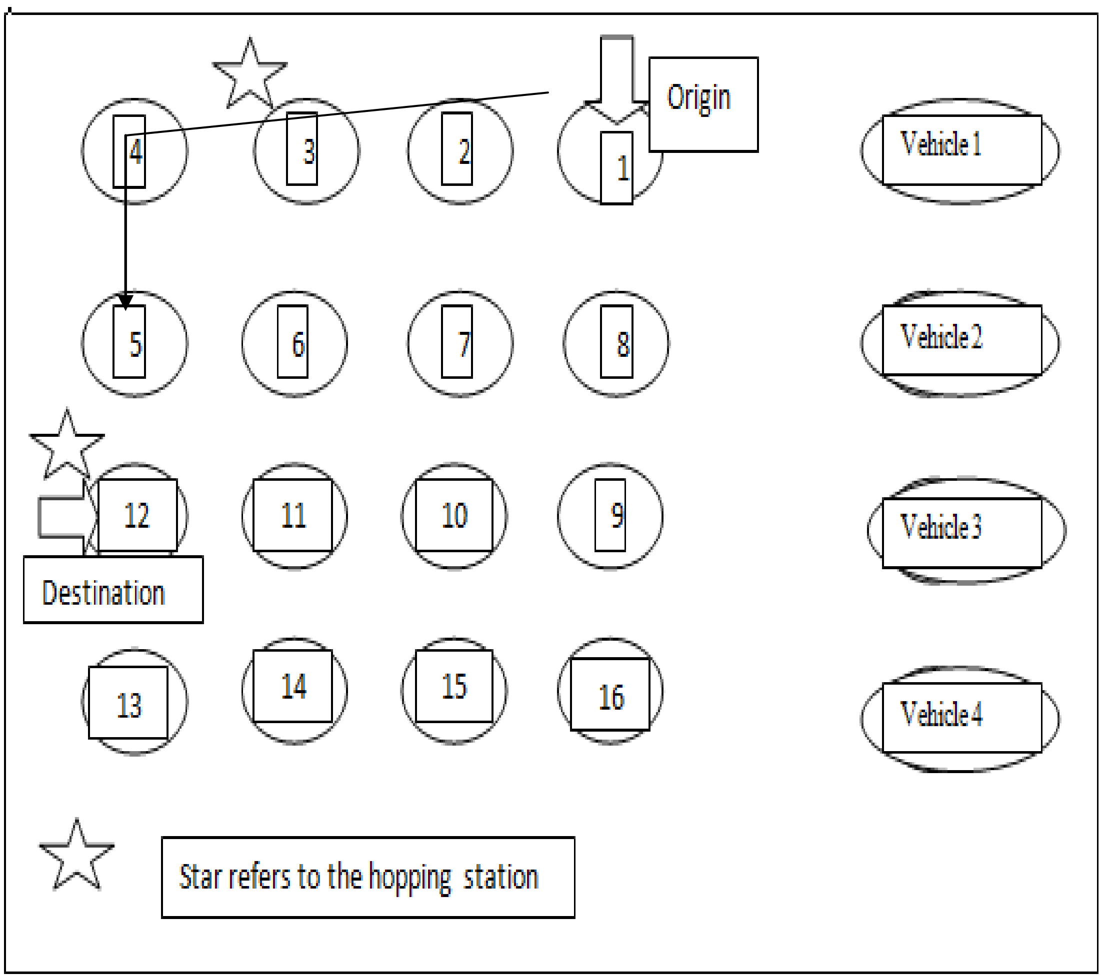


Figure 4.4 Path selected by Node (Vehicle 2) to travel from source to destination

Here in the figure 4.4, Vehicle 2 is traveling from the origin to destination and the vehicle surpasses the vertex 3 which is bearing some information that needs to be transmitted. When the vehicle reaches the vertex 3, the inbuilt GPS system is evaluated by the node, and the path that is selected by the vehicle is taken into consideration.

Since the path selected by the vehicle does not match with the path that has been selected by the End to End delivery vertex the information is not hopped and wait for next vehicle is done.

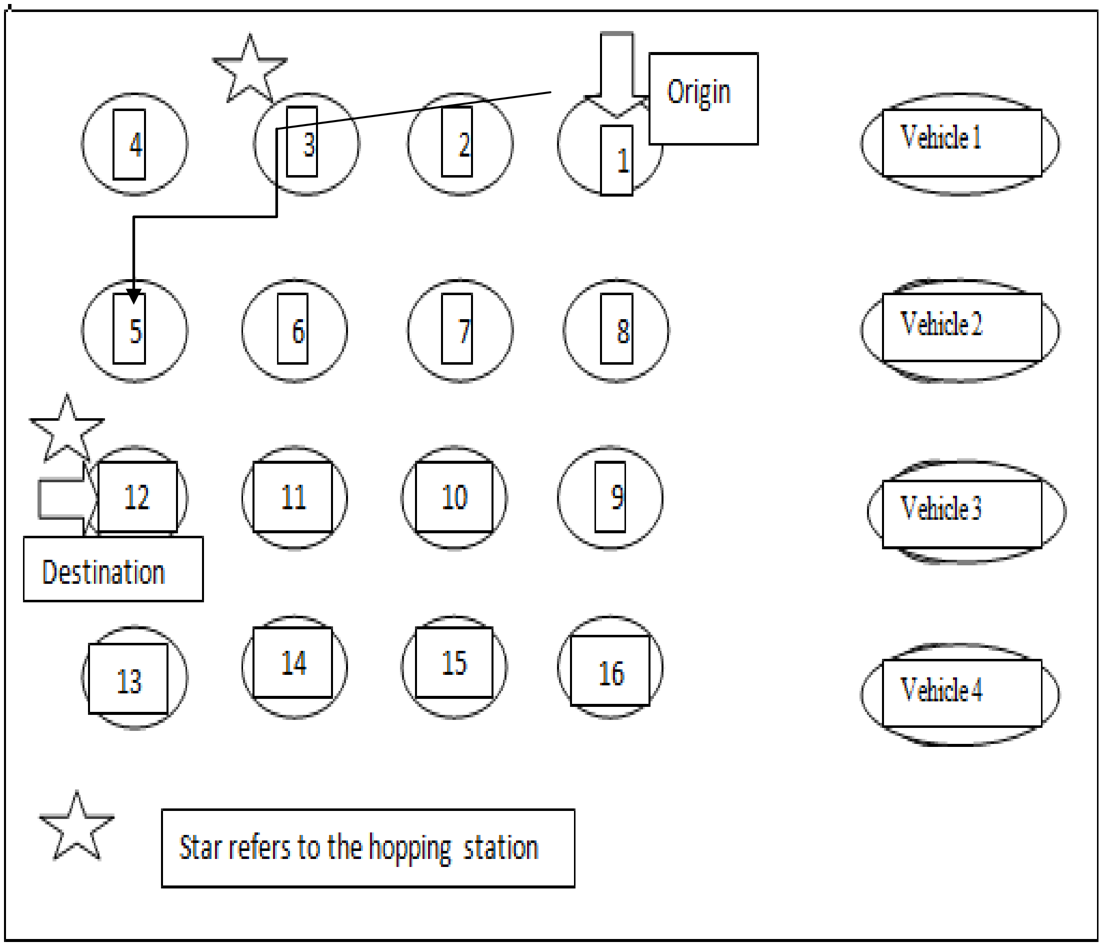


Figure 4.5 Path selected by Node (Vehicle 3) to travel from source to destination

Here in the figure 4.5 Vehicle 3 is traveling from the origin to destination and the vehicle surpasses the vertex 3 which is bearing some information that needs to be transmitted. When the vehicle reaches the vertex 3, the inbuilt GPS system is evaluated by the node, and the path that is selected by the vehicle is same as per the requirement of End to End delivery node, hence the vehicle is taken into consideration and the information on the vertex 3 is hopped on to the vehicle, which bears the information transfers from one vertex to another and achieves the objective of the algorithm.

Once the information reaches the target vertex, the information from the vehicle is once again hopped onto the destination, where there is no more future aspiration from the vehicle is required.

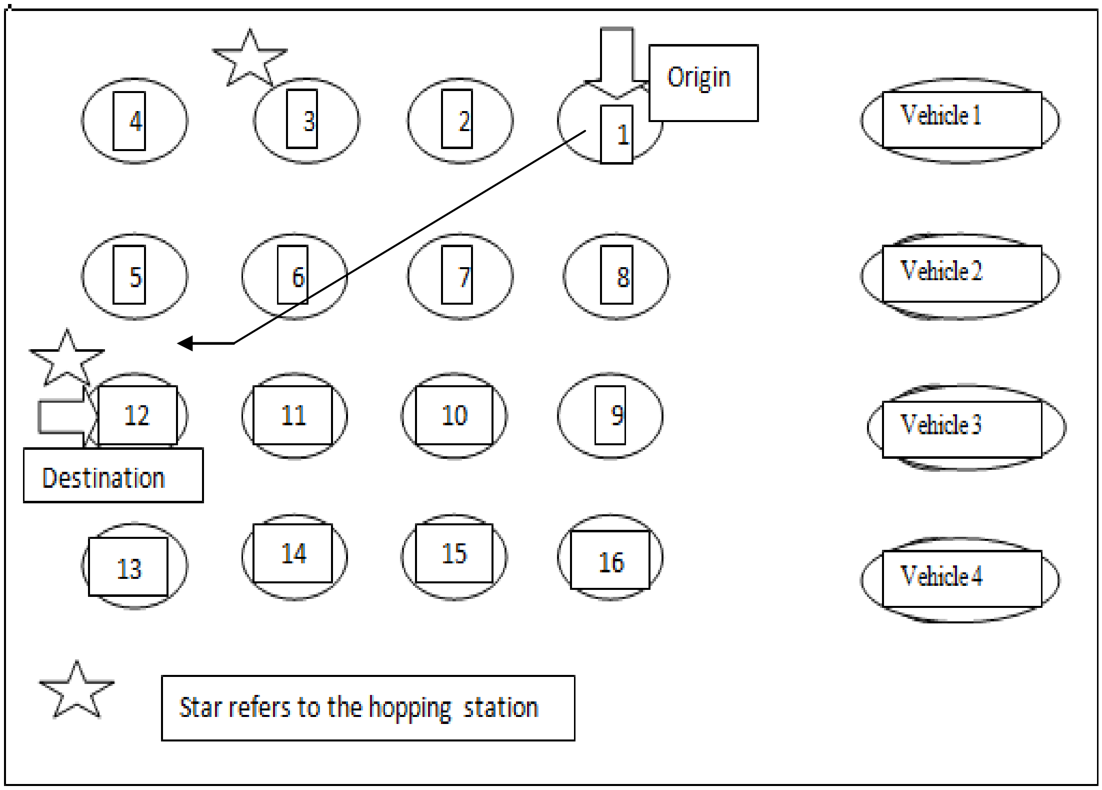


Figure 4.6 Path selected by Node (Vehicle 4) to travel from source to destination

Here in the figure 4.6 Vehicle 4 is traveling from the origin to destination and the vehicle does not surpasses the vertex 3 which is bearing some information that needs to be transmitted. When the vehicle is not even on the way of transmitting vertex, there is no evaluation done for the path, as the vehicle does not even satisfy the initial condition of the TCDD.

Conclusion

After introducing the proposed system, we have quantified that the existing system if may work as programmed and also incorporate the physical availabilities i.e. GPS system conclude in synchronization with the existing system. When we further evaluate it is concluded that not only the objective of Data Latency is eliminated rather an additional feature of enhanced delivery ratio may also be achieved in a significant manner.

Chapter 5

Result

Introduction

This is the phase where the proposed system is given a test run and final outcomes for the proposed work is evaluated on different parameters. However there are two different things that are of much importance and needs to be emphasized. First is to identify the parameters of evaluation and second is to identify the environment of the testing, as the aim of the entire process is not only to make sure that the program is being executed, but also that it has overcome the expected results on realistic environment. In the chapter we will be emphasizing on the result based achievements by comparing them in various inputs.

5.1 Results

Until now the entire operation has been accepted on evaluation software and has had a successful run so far. But we may never consider the successful execution of the program as an achievement for final reviewing until it surpasses some necessary conditions that are needed to be achieved, for the final acceptance of the task performed. The entire process of evaluation has been done on Network Simulator -2 which has fabricated the results in mainly three parameters:

1. Data Latency (ms)
2. Connection
3. Packet Delivery Ratio (%)

Before we go ahead with discussing the evaluation on results, let us have a basic understanding about the procedure of software execution and also acknowledge the range in which the software will be producing output. Hence to test the authentication of the proposed model we had been working upon, we make use of Network Simulator 2 (NS2). Let us take some time to understand the conceptual working of the testing software.

5.2 Network Simulator

Network Simulator (NS) is a name for series of discrete event network simulators, specifically Network Simulator -1, Network Simulator - 2 and Network Simulator -3. All of them are discrete-event network simulator, primarily used in research] and teaching. It is free software, publicly available under the GNU GPLv2 license for research, development and use.

The goal of the ns-3 project is to create an open simulation environment for networking research that will be preferred inside the research community:

1. It should be aligned with the simulation needs of modern networking research.
2. It should encourage community contribution, peer review, and validation of the software.

Since the process of creation of a network simulator that contains a sufficient number of high-quality validated, tested and actively maintained models requires a lot of work, ns-3 project spreads this workload over a large community of users and developers.

5.3 Network Simulator Application Development Lifecycle

Network Simulator - 1

The first version of Network Simulator, known as NS-1, which was developed at Lawrence Berkeley National Laboratory (LBNL) in the 1995-97 timeframe by Steve McCanne, Sally Floyd, Kevin Fall, and other contributors. This was known as the LBNL Network Simulator, and derived from an earlier simulator known as REAL by S. Keshav. The core of the simulator was written in C++, with TCL-based scripting of simulation scenarios.

Network Simulator - 2

In 1996-97, Network Simulator version 2 (Network Simulator -2) was initiated based on a refactoring by Steve McCanne. Use of TCL was replaced by MIT's Object TCL (OTCL), an object-oriented dialect of TCL. The core of Network Simulator – 2 is also written in C++, but the C++ simulation objects are linked to shadow objects in Tcl and variables can be linked between both language realms. Simulation scripts are written in the OTCL language, an extension of the TCL scripting language.

Presently, Network Simulator-2 consists of over 300,000 lines of source code, and there is probably a comparable amount of contributed code that is not integrated directly into the main distribution (many forks of ns-2 exist, both maintained and unmaintained). It runs on GNU/Linux, FreeBSD, Solaris, Mac OS X and Windows versions that support Cygwin. It is licensed for use under version 2 of the GNU General Public License.

5.4 Authentication of program

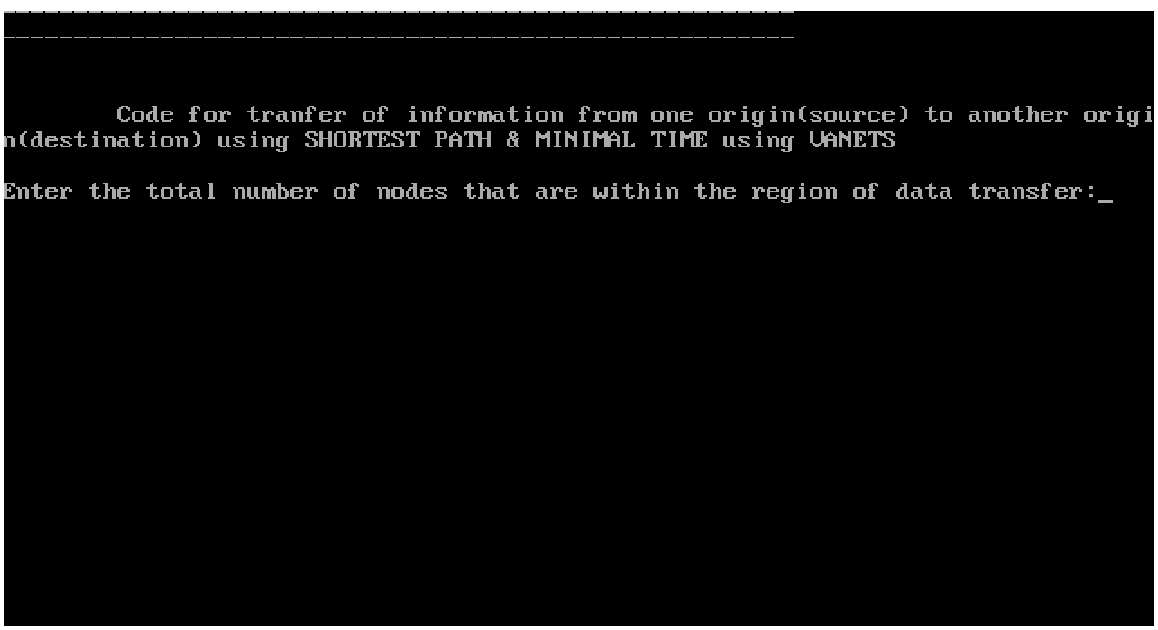


Fig 5.1 Showing the output of the program

The development of the coding is done in a flexible manner that the code can easily switch over to a maximum range of 70 nodes at one time and can do a bit wise comparison of each node according to assigned weights on the path of all the nodes within a region and can depict an optimal path for the vehicle to travel with an objective of faster delivery of information from one point of the entire network to another.

Flexibility in the algorithm has been adapted that the executioner strategy may only adopt the node that is going on the desired direction as per the prompt transfer for information. This in turn eliminates the basic constraints of delay of data delivery, an optimal observation of the current location of data by synchronizing the GPS system of the vehicle. The add on variable that we get with this kind of functioning is that we are able to identify the Packet Delivery Ratio, by terms of which we can easily identify the success rate of delivery of information as it entirely depends upon the arrival/passing by of a vehicle to the destination of delivery.

Here the requirement is not that the vehicle should be stationary at the point of delivery. At delivery node, a vertex has a radar device which identifies the address of the packet and if the packet of destined address it may be transferred to the informational radar, else it is ignored and simultaneously the vehicle passes on the destination it is headed.

Following are the various screen shots of outputs received from the program.

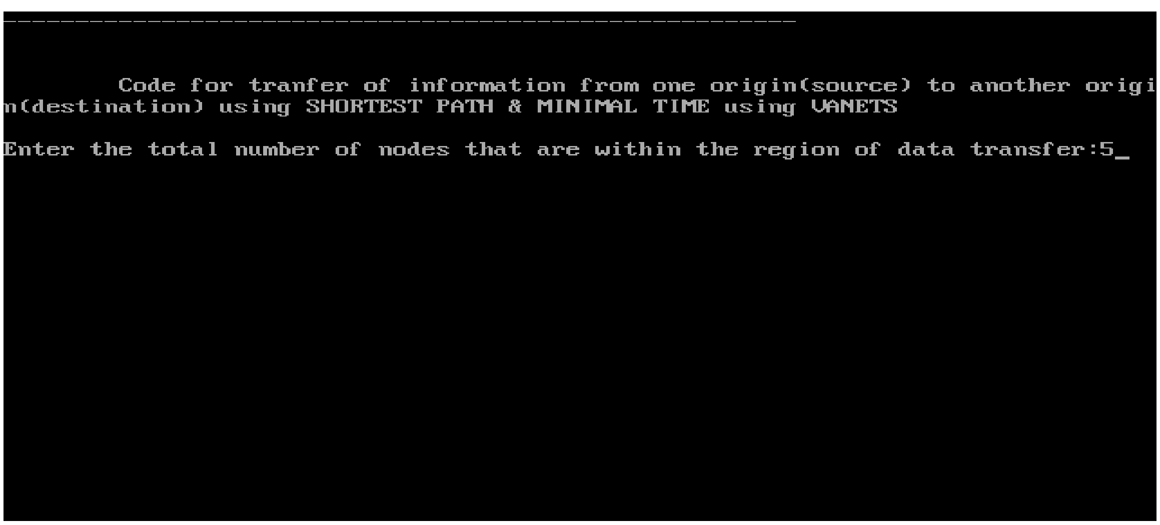


Fig 5.2 Entering of Vertex nodes in the Network Area

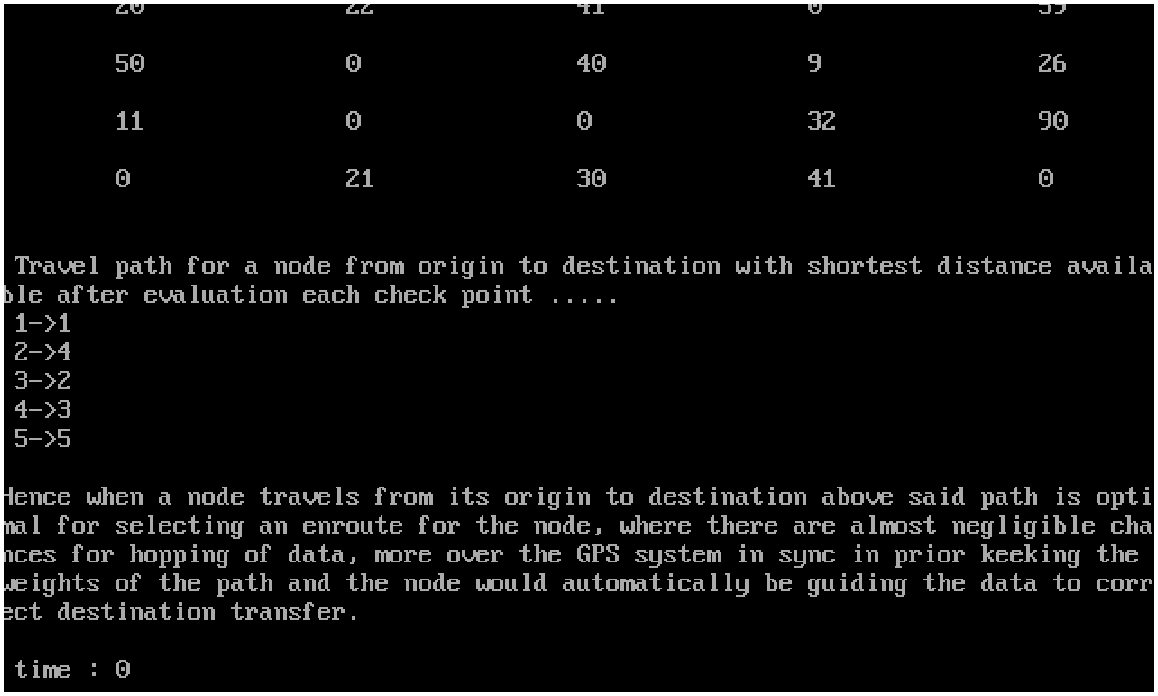


Fig 5.3 Program Output

5.5 Evaluation of Nodes

In concern to the defined problem an algorithm and evaluation process was defined, on which we tested on a different range of values varying from a minimum node of 5 to 70 in number. Also while execution of the program total time lapsed to execute nodes have also been calculated. A variance of time bound output have been calculated at two different outputs by assigning different weights at nodes each time of execution.

Resultant to it, two times the test have been calculated for the same number of nodes bearing different values and have obtained the resultant values for the time taken to execute a program in milliseconds. Following table shows the variance of time that has been taken by the program to execute. A glimpse of it would define the fractional difference between the two.

Apart from tabular form the information has also been depicted graphically to have a better evaluation of difference of results in two different graph tables.

Sr. No	**No of Nodes**	**Time Consumption (in milliseconds)**	**Time Consumption (in milliseconds)**
		Inputs I	**Inputs II**
1	5	0:021	0:021
2	10	0:021	0:021
3	15	0:026	0:025
4	20	0:027	0:024
5	25	0:029	0:026
6	30	0:030	0:029
7	35	0:032	0:030
8	40	0:033	0:031
9	45	0:035	0:033
10	50	0:037	0:036
11	55	0:038	0:037
12	60	0:040	0:039
13	65	0:043	0:042
14	70	0:046	0:044

Table 5.1 Depicting the number of nodes and the variation of time taken to evaluate the number of nodes along with execution time.

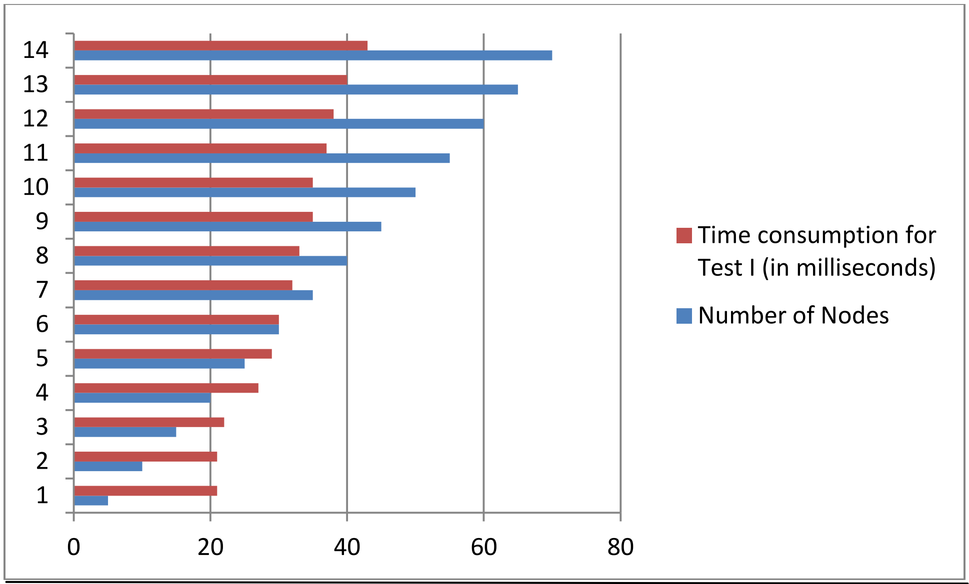


Figure 5.4 Evaluation of Test I conducted on nodes

In the above table bar chart (Fig. 5.4) we see the change in variance values of the time taken to evaluate a set of nodes for calculation of the optimal path that a vehicle should take to travel from one source to another or may be destination. We observe that as the number of nodes increases the complexity of the evaluation process also increases however the entire process is not subjective to the increasing in the number of nodes, it also is evaluated on the basis of the range of weights that we assign to each node, which is perhaps more easy to calculate in terms of latency in regard to the time taken by the node to transfer the information from one location to another.

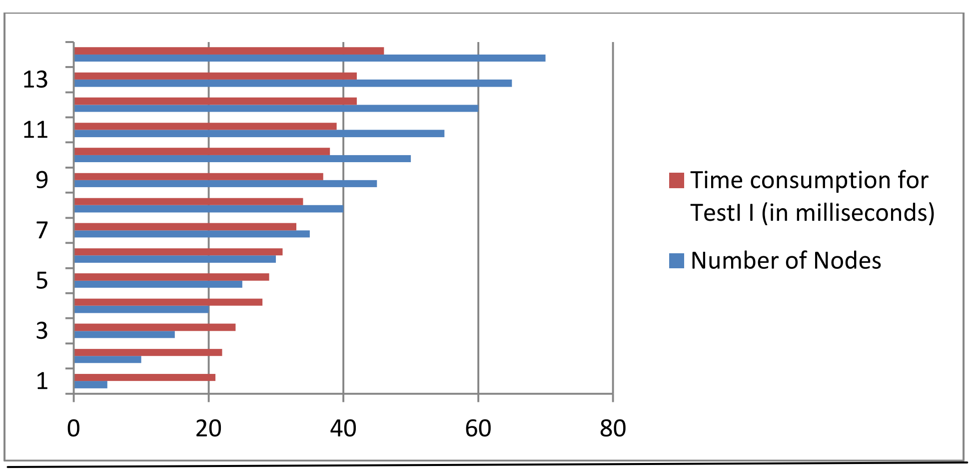


Figure 5.5 Evaluation of Test II conducted on nodes

In the above table bar chart (Fig. 5.5) we see the change in variance values of the time taken to evaluate a set of nodes for calculation of the optimal path that a vehicle should take to travel from one source to another or may be destination. However when we compare both the tables, we see that there is very minute change in the time based evaluation. However this has not been evaluated on the basis of same input as given in Test I. A very minute variation in terms of input has been there when we had assigned the values to nodes.

To conclude the tests based on time constraint we may sum up by depicting our results by quoting that the time complexity would increase by assigning higher values to the nodes in case the number of nodes are high, and would significantly drop if the assigned values to the nodes are less, however number of nodes being high. So there is a parallel run between the number of nodes and the weights at each node.

5.6 Data Latency

Latency is a time interval between the stimulation and response, or, from a more general point of view, as a time delay between the cause and the effect of some physical change in the system being observed. Latency is physically a consequence of the limited velocity with which any physical interaction can propagate. This velocity is always lower than or equal to the speed of light. Therefore every physical system that has spatial dimensions different from zero will experience some sort of latency, regardless of the nature of stimulation that it has been exposed to.

The precise definition of latency depends on the system being observed and the nature of stimulation. In communications, the lower limit of latency is determined by the medium being used for communications. In reliable two-way communication systems, latency limits the maximum rate that information can be transmitted, as there is often a limit on the amount of information that is "in-flight" at any one moment. In the field of human–machine interaction, perceptible latency has a strong effect on user satisfaction and usability.

Since we are referring to transfer of information from one part of the network to another, we would first need to understand the exact terminology of Network Latency as well.
Network latency in a packet-switched network is measured either one-way (the time from the source sending a packet to the destination receiving it), or round-trip delay time (the one-way latency from source to destination plus the one-way latency from the destination back to the source). Round-trip latency is more often quoted, because it can be measured from a single point. Note that round trip latency excludes the amount of time that a destination system spends processing the packet. Many software platforms provide a service called ping that can be used to measure round-trip latency. Ping performs no packet processing; it merely sends a response back when it receives a packet (i.e. performs a no-op), thus it is a first rough way of measuring latency. Ping cannot perform accurate measurements, principally because it uses the ICMP protocol that is used only for diagnostic or control purposes, and differs from real communication protocols such as TCP. Furthermore routers and ISP's might apply different traffic shaping policies to different protocols.

However, in a non-trivial network, a typical packet will be forwarded over many links via many gateways, each of which will not begin to forward the packet until it has been completely received. In such a network, the minimal latency is the sum of the minimum latency of each link, plus the transmission delay of each link except the final one, plus the forwarding latency of each gateway. In practice, this minimal latency is further augmented by queuing and processing delays. Queuing delay occurs when a gateway receives multiple packets from different sources heading towards the same destination. Since typically only one packet can be transmitted at a time, some of the packets must queue for transmission, incurring additional delay. Processing delays are incurred while a gateway determines what to do with a newly received packet. A new and emergent behavior called buffer bloat can also cause increased latency that is an order of magnitude or more. The combination of propagation, serialization, queuing, and processing delays often produces a complex and variable network latency profile.

In simulation applications, 'latency' refers to the time delay, normally measured in

milliseconds (1/1,000 sec), between initial input and an output clearly discernible to the simulator trainee or simulator subject. Latency is sometimes also called transport delay.
Some authorities distinguish between latency and transport delay by using the term 'latency' in the sense of the extra time delay of a system over and above the reaction time of the vehicle being simulated, but this requires a detailed knowledge of the vehicle dynamics and can be controversial.

Importance of Motion and Visual Latencies

In simulators with both visual and motion systems, it is particularly important that the latency of the motion system not be greater than of the visual system, or symptoms of simulator sickness may result. This is because in the real world, motion cues are those of acceleration and are quickly transmitted to the brain, typically in less than 50 milliseconds; this is followed some milliseconds later by a perception of change in the visual scene. The visual scene change is essentially one of change of perspective and/or displacement of objects such as the horizon, which takes some time to build up to discernible amounts after the initial acceleration which caused the displacement. A simulator should therefore reflect the real-world situation by ensuring that the motion latency is equal to or less than that of the visual system and not the other way round. Since we are refereeing to the fact based data evaluation, we have integrated the time based evaluation of nodes on to Network Simulator 2 and had calculated latency for different nodes.

Data latency can also be catered in terms of either a packet being late in transmission, or further interconnected packets that first wait for the header packet to be received or later remaining packets to be received at time. Here the motion sensor takes time to build up the discernible amounts after the initial acceleration that actually causes displacement.

The automation of the displacement sensor is so advanced that it can measure the displacement value in milliseconds (1/1,000 sec).

Sr. No	No of Nodes	Latency (ms)
1	5	25.25
2	10	23.191
3	15	22.132
4	20	21.073
5	25	20.014
6	30	20.006
7	35	19.45
8	40	19.32
9	45	18.78
10	50	18.531
11	55	15.642
12	60	13.535
13	65	10.428
14	70	9.321

Table 5.2 Values of nodes with Data Latency in milliseconds.

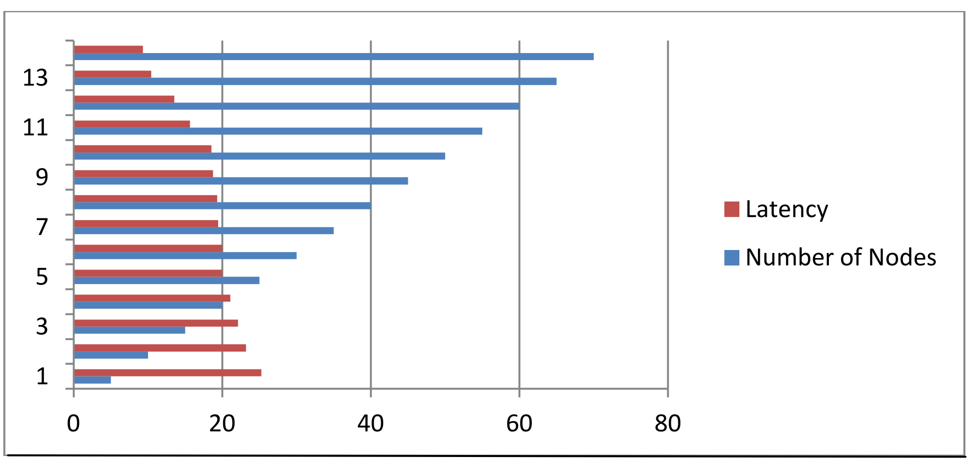


Figure 5.6 Evaluation results of Data Latency

5.7 Evaluation for Connection

Connectivity is one of the basic concepts of graph theory: it asks for the minimum number of elements (nodes or edges) that need to be removed to disconnect the remaining nodes from each other. It is closely related to the theory of network flow problems. The connectivity of a graph is an important measure of its robustness as a network.

In an undirected graph G, two vertices u and v are called connected if G contains a path from u to v. Otherwise, they are called disconnected. If the two vertices are additionally connected by a path of length 1, i.e. by a single edge, the vertices are called adjacent. A graph is said to be connected if every pair of vertices in the graph is connected.

A connected component is a maximal connected sub graph of G. Each vertex belongs to exactly one connected component, as does each edge.

A directed graph is called weakly connected if replacing all of its directed edges with undirected edges produces a connected (undirected) graph. It is connected if it contains a directed path [26] from u to v or a directed path from v to u for every pair of vertices u, v. It is strongly connected or strong if it contains a directed path from u to v and a directed path from v to u for every pair of vertices u, v. The strong components are the maximal strongly connected sub graphs.

A cut, vertex cut, or separating set of a connected graph G is a set of vertices whose removal renders G disconnected. The connectivity or vertex connectivity κ(G) (where G is not a complete graph) is the size of a minimal vertex cut. A graph is called k-connected or k-vertex-connected if its vertex connectivity is k or greater. This means a graph G is said T Analogous [22] concepts can be defined for edges. In the simple case in which cutting a single, specific edge would disconnect the graph, that edge is called a bridge. More generally, the edge cut of G is a group of edges whose total removal renders the graph disconnected. The edge-connectivity λ(G) is the size of a smallest edge cut, and the local edge-connectivity λ(u,v) of two vertices u, v is the size of a smallest edge cut disconnecting u from v. Again, local edge-connectivity is symmetric. A graph is called k-edge-connected if its edge connectivity is k or greater.

The problem of determining whether two vertices in a graph are connected can be solved efficiently using a search algorithm, such as breadth-first search. More generally, it is easy to determine computationally whether a graph is connected (for example, by using a disjoint-set data structure), or to count the number of connected components. A simple algorithm might be written in pseudo-code as follows:

1. Begin at any arbitrary node of the graph, G
2. Proceed from that node using either depth-first or breadth-first search, counting all nodes reached.
3. Once the graph has been entirely traversed, if the number of nodes counted is equal to the number of nodes of G, the graph is connected; otherwise it is disconnected.

Keeping the merging theorem in mind, we had simulated the test based on connectivity onto Network Simulator 2 to find the level of connection between the total no of nodes and obtained the following results.

Sr. No	No. of Nodes	Connection
1	5	27
2	10	32
3	15	35
4	20	38
5	25	42
6	30	45
7	35	49
8	40	50
9	45	56
10	50	58
11	55	63
12	60	66
13	65	70
14	70	73

Table 5.3 Values of nodes with inter connectivity within a network

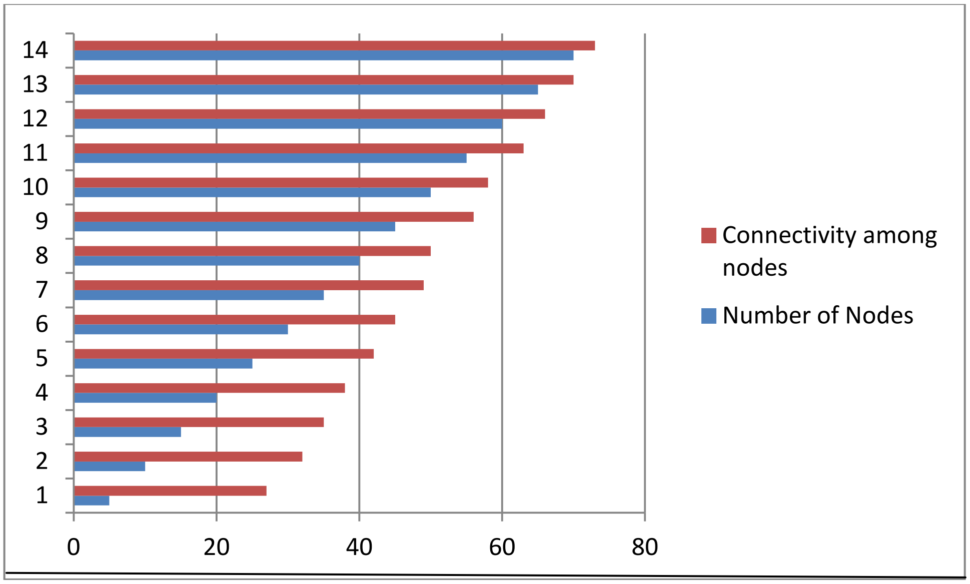


Figure 5.7 Evaluation results of Connectivity among nodes

5.8 Packet Delivery Ratio

Packet delivery ratio is reffered as the ratio of the number of delivered data packet to the destination. This illustrates the level of delivered data to the destination.

$$\sum \text{Number of packet received} / \sum \text{Number of packet sent}$$

The greater value of packet delivery ratio means the better performance of the protocol. There is one standard concern that is taken in consideration for packet delivery ratio i.e. End-to-end Delay.

In the process of testig the simulation on the existing facts, analyzation has been done on Network Simulator – 2 and had obtained in the following table are on the basis of the data informational refencement of behaviour among nodes when information travels on the node and the infomration is being transferred from one part of the network to another. Here the ratio is also considered on the basis of packets initnally obtained by the vehicle and the number of packets that are delived at the destination host, where packes are counted.

Sr. No	No. of Nodes	Packet Delivery Ratio
1	5	53.52
2	10	62.6
3	15	71.68
4	20	80.76
5	25	82.42
6	30	84.08
7	35	85.74
8	40	87.4
9	45	89.06
10	50	90.72
11	55	92.38
12	60	94.04
13	65	95.7
14	70	97.36

Table 5.3 Values of Data Connectivity Ratio of Packet Delivery Ratio

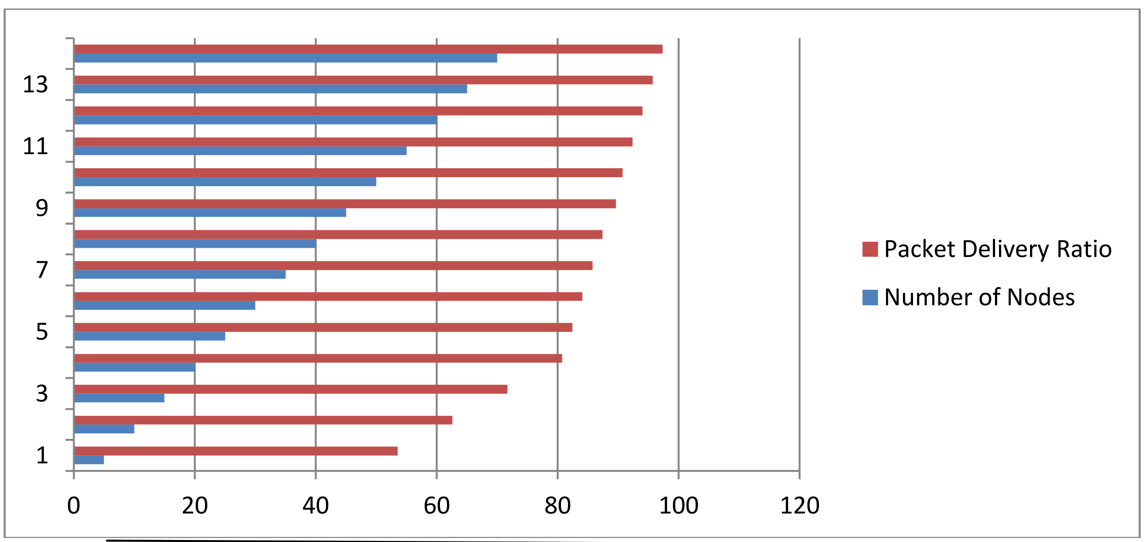


Figure 5.8 Evaluation results of Packet Delivery Ratio of information from one part of the network to another

5.9 Results Evaluation

The basic objective to the overall process was to calculate the the average time taken by a data packet to arrive at the destination. The most implicit concerns that are needed to adher is the delay caused to the infomration in route is the discovery process and the queue in data packet transmission [31]. Only the data packets that successfully delivered to destinations that counted. It can be calculated by the formulation:

$$\sum \text{(arrive time – send time)} / \sum \text{Number of connections}$$

The lower value of end to end delay means the better performance of the protocol.
The other concerned terminology is the process is Packet Lost.

Packet Lost : the total number of packets dropped during the simulation.

It can be calculated by a basic formula:
Packet lost = Number of packets sent – Number of packets received

The lower value of the packet lost means the better performance of the protocol.

5.10 Throughput Comparisons

We know that throughput increases when connectivity is better. Hence let us review the output of different parameters in vivid prospective.

5.10.1 **Throughput : -** It is defined as the total number of packets delivered over the total simulation time. The throughput comparison shows that the three algorithms performance margins are very close under traffic load of 50 and 100 nodes in MANET scenario and have large margins when number of nodes increases to 200.
Mathematically, it can be defined as:
Throughput = N/1000

Where N is the number of bits received successfully by all destinations.

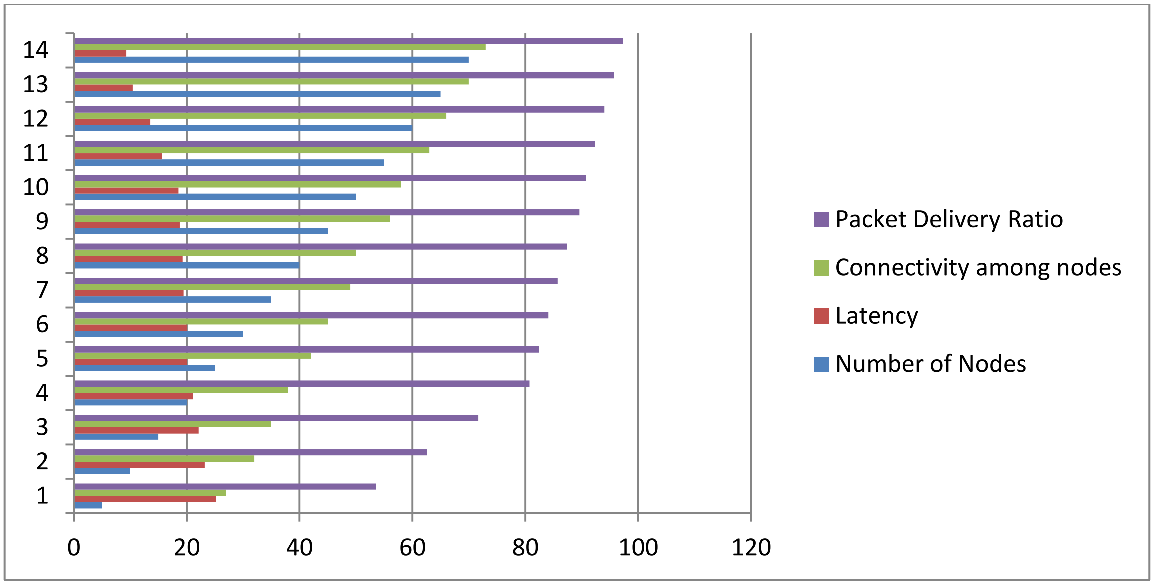


Figure 5.9 Combined evaluation of Data Latency, Connection and Packet Delivery Ratio comparison

Above graphs shows the variation in Data Latency, Connection and Packet Delivery Ratio among different a network. We analyze from the following output that irrespective of the number of nodes being enhancing by subsequent test, the more the number of nodes, the better the interconnectivity among them.

5.10.2 End to End Delay

The average time it takes a data packet to reach the destination. This includes all possible delays caused by buffering during route discovery latency, queuing at the interface queue. This metric is calculated by subtracting time at which first packet was transmitted by source from time at which first data packet arrived to destination. Mathematically, it can be defined as: Avg. EED=S/N Where S is the sum of the time spent to deliver packets for each destination, and N is the number of packets received by the all destination nodes.

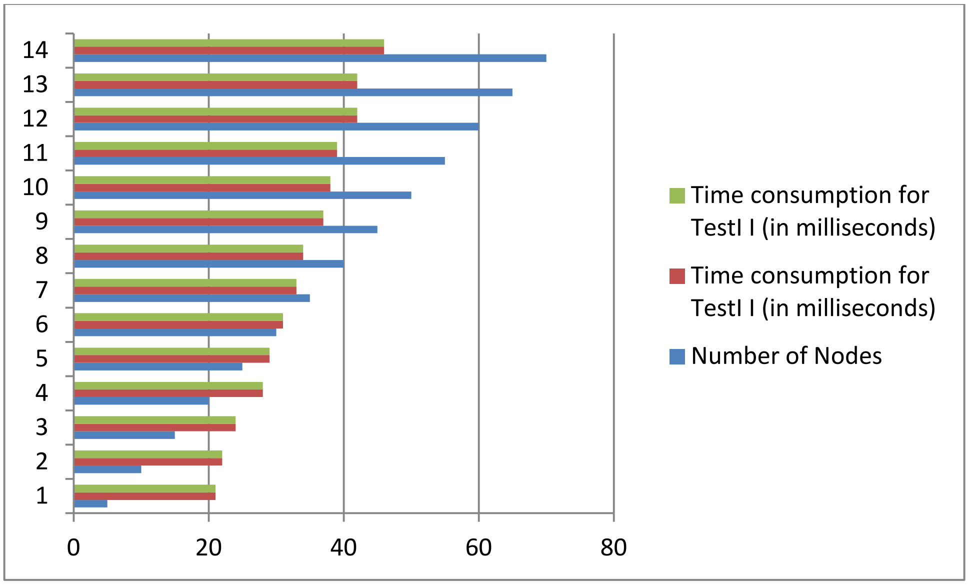


Fig 5.10 Combined evaluation of Time Test on nodes with different weights.

Hence when we calculate the average of the total time taken to accomplish a task during the transfer of information varying from Test 1 to Test 2, we accumulate to a resultant value that there is a total difference of fractional minutes that constitute to the variation of tests according to the values assigned on the nodes. The average may vary from time to time and would constitute on the value of variables given to individual nodes given on a network path.

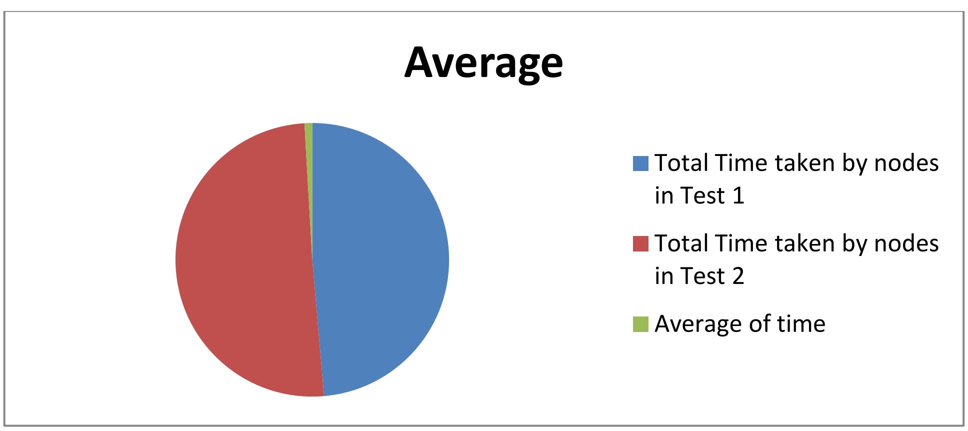


Figure 5.11 Total time calculation of Test Conditions

Conclusion

At the end of the chapter we conclude that there are various assumptions that can be carried out on the transfer of data, however the optimal mode of transfer is to primarily evaluate the available network and weights of different edges that lie in a path. There is also a need to understand the self requirement before performing operation and later work on a guided medium so that there can be minimal amount of data loss and delay no matter what so may happen.

Chapter 6
Conclusion and Future Work

6.1 Conclusion

This paper proposes a Trajectory Controlled Data Dissemination (TCDD) Algorithm for VANETs, tailored for the data forwarding for roadside reports in light-traffic vehicular ad hoc networks. State-of-the-art schemes have demonstrated the effectiveness of their data forwarding strategies by exploiting known vehicular traffic activity (e.g., destination and speed). These results are encouraging and fruitful. This paper presents the first attempt to effectively utilize vehicles' trajectory information in a privacy-preserving manner. In our design, such trajectory information is combined with the vehicular traffic activity by synchronizing with the GPS system preinstalled in the system for assuring correct data transfer of information from one part of the network to another part. Similarly in a distributed way, each individual vehicle computes its end-to-end expected delivery by selecting the shortest path available from the source to destination. For the accurate End-to-End delay computation, this paper also proposes three fundamental aspects while data transfer i.e data latency, connection and delivery aspect ratio to estimate the exact time and delivery of information (packet) to destination node. Theoretical analysis and extensive simulation has been adopted to acquire optimal results. It is shown that more the number of nodes within a network, better is the interconnectivity among nodes

6.2 Future Work

Future opportunities could explore following:

- The proposed framework has been implemented keeping nodes within a network in mind; however another alternative research work can be carried out to transfer of information among various vehicles that are traveling in uni-direction on the basis speed at which they travel towards the target node.
- Security protocol can be incorporated with the proposed solution to encrypt the data before the information is to be hopped on the desired vehicle which is going towards the destination side.

References

[1] Ma, Chunmei, and Nianbo Liu. "Traffic-aware data delivery scheme for urban vehicular sensor networks." *International Journal of Distributed Sensor Networks 2013* (2013)

[2] Lochert, Christian, Björn Scheuermann, Murat Caliskan, and Martin Mauve. "The feasibility of information dissemination in vehicular ad-hoc networks." *In Wireless on Demand Network Systems and Services, 2007. WONS'07. Fourth Annual Conference on, p. 92-99. IEEE* (2007)

[3] Gu, Jaehoon Jeong Shuo Guo Yu, and D. Tian He Du. "tbd: trajectory-based data forwarding for light-traffic vehicular networks." *Distributed Computing* (2009)

[4] Ding, Yong, and Li Xiao. "SADV: static-node-assisted adaptive data dissemination in vehicular networks." *Vehicular Technology, IEEE Transactions on 59, no. 5* (2010)

[5] Eckhoff, David, Christoph Sommer, Reinhard German, and Falko Dressler. "Cooperative awareness at low vehicle densities: How parked cars can help see through buildings." In *Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE*, pp. 1-6. IEEE, (2011)

[6] Zheng, Zizhan, Zhixue Lu, Prasun Sinha, and Santosh Kumar. "Maximizing the contact opportunity for vehicular internet access." *In INFOCOM, 2010 Proceedings IEEE, pp. 1-9. IEEE* (2010)

[7] Wu, Celimuge, Satoshi Ohzahata, and Toshihiko Kato. "Joint MAC network layer broadcast protocol for vehicular ad hoc networks." *Communications Workshops (ICC), 2013 IEEE International Conference on. IEEE* (2013)

[8] Salazar Hernández, Dagoberto José, Manuel Hernández Pajares, José Miguel Juan Zornoza, and Jaume Sanz Subirana. "The GPS Toolkit: world class open source software tools for the GNSS research community." *7a Setmana Geomàtica* (2008)

[9] Palomar, Esther, et al. "Hindering false event dissemination in VANETs with proof-of-work mechanisms." *Transportation Research Part C: Emerging Technologies 23 p 85-97* (2012)

[10] Ali, G. G., Edward Chan, and Wenzhong Li. "Supporting real-time multiple data items query in multi-RSU vehicular ad hoc networks (VANETs)." *Journal of Systems and Software* 86, no. 8 (2013)

[11] SONG, Chao, Ming LIU, Hai-gang GONG, Gui-hai CHEN, and Jian-nong CAO. "Utilizing the dropped packets for data delivery in VANETs." *The Journal of China Universities of Posts and Telecommunications* 20, no. 3 (2013)

[12] Bitam, Salim, Abdelhamid Mellouk, and Sherali Zeadally. "HyBR: A Hybrid Bio-inspired Bee swarm Routing protocol for safety applications in Vehicular Ad hoc NETworks (VANETs)." *Journal of Systems Architecture* 59, no. 10 (2013)

[13] Dong, Wei, Chun Chen, Xue Liu, Guodong Teng, Jiajun Bu, and Yunhao Liu. "Bulk data dissemination in wireless sensor networks: Modeling and analysis." *Computer Networks* 56, no. 11 (2012)

[14] Chen, Tao, Deke Guo, Yuan He, Honghui Chen, Xue Liu, and Xueshan Luo. "A Bloom filters based dissemination protocol in wireless sensor networks." *Ad Hoc Networks* 11, no. 4 (2013)

[15] Ruiz, Patricia, Bernabé Dorronsoro, Pascal Bouvry, and Lorenzo Tardón. "Information dissemination in VANETs based upon a tree topology." *Ad Hoc Networks 10, no. 1* (2012)

[16] Schwartz, Ramon S., Anthony E. Ohazulike, Christoph Sommer, Hans Scholten, Falko Dressler, and Paul Havinga. "On the applicability of fair and adaptive data dissemination in traffic information systems." *Ad hoc networks 13* (2014)

[17] Annu Mor ,"Study of Different Type of Data Dissemination Strategy in VANET.", *International Journal of Engineering Science and Innovative Technology* (2012)

[18] Ye, Fei, Sumit Roy, and Haobing Wang. "Efficient data dissemination in vehicular ad hoc networks." *Selected Areas in Communications, IEEE Journal on* 30, no. 4 (2012)

[19] Boukerche, Azzedine, Horacio ABF Oliveira, Eduardo F. Nakamura, and Antonio AF Loureiro. "Vehicular ad hoc networks: A new challenge for localization-based systems." *Computer communications 31, no. 12* (2008)

[20] Kumar, Narendra, Chandrika Swamy, and K. N. Nagadarshini. "Efficient Garbage Disposal Management in Metropolitan Cities Using VANETs." *Journal of Clean Energy Technologies 2, no. 3* (2014)

[21] Salkuyeh, Mostafa Asgharpoor, Faramarz Hendessi, and T. Aaron Gulliver. "Data dissemination with rateless coding in a grid vehicular topology." *EURASIP Journal on Wireless Communications and Networking* 2013, no. 1 (2013)

[22] Gong, Haigang, Nianbo Liu, Lingfei Yu, and Chao Song. "An Efficient Data Dissemination Protocol with Roadside Parked Vehicles' Assistance in Vehicular Networks." *International Journal of Distributed Sensor Networks* 2013 (2013)

[23] Nadeem, Tamer, Pravin Shankar, and Liviu Iftode. "A comparative study of data dissemination models for VANETs." In *Mobile and Ubiquitous Systems-Workshops, 2006. 3rd Annual International Conference on*, pp. 1-10. IEEE, (2006)

[24] Ali, GG Md Nawaz, Edward Chan, and Wenzhong Li. "On scheduling data access with cooperative load balancing in vehicular ad hoc networks (VANETs)." *The Journal of Supercomputing 67, no. 2* (2014)

[25] Rohal, Pankaj, Ruchika Dahiya, and Prashant Dahiya. "Study and analysis of throughput, delay and packet delivery ratio in manet for topology based routing protocols (aodv, dsr and dsdv)." *International journal for advance research in engineering and technology 1* (2013)

[26] Abuelela, M. Olariu, and I. S Stojmenovic. "Opera: Opportunistic packet relaying in disconnected vehicular ad hoc networks." *Mobile Ad Hoc and Sensor Systems, 2008. MASS 2008. 5th IEEE International Conference* (2008)

[27] Lee, Youndo, Hojin Lee, Nakjung Choi, Yanghee Choi, and Taekyoung Kwon. "Macro-level and Micro-level Routing (MMR) for urban vehicular ad hoc networks." In

Global Telecommunications Conference, 2007. GLOBECOM'07. IEEE, pp. 715-719. IEEE, (2007)

[28] Lee, Hojin, Youndo Lee, Taekyoung Kwon, and Yanghee Choi. "Virtual Vertex Routing (VVR) for Course-Based Vehicular Ad Hoc Networks." In *WCNC*, pp. 4405-4410. (2007)

[29] Rodriguez, Pablo, Rajiv Chakravorty, Julian Chesterfield, Ian Pratt, and Suman Banerjee. "Mar: A commuter router infrastructure for the mobile internet." In *Proceedings of the 2nd international conference on Mobile systems, applications, and services*, pp. 217-230. ACM, (2004)

[30] Abuelela, M. Olariu, and I. S Stojmenovic. "Opera: Opportunistic packet relaying in disconnected vehicular ad hoc networks." Mobile Ad Hoc and Sensor Systems, 2008. MASS 2008. 5th IEEE International Conference, (2008)

[31] Nadeem, Tamer, Sasan Dashtinezhad, Chunyuan Liao, and Liviu Iftode. "TrafficView: traffic data dissemination using car-to-car communication." *ACM SIGMOBILE Mobile Computing and Communications Review* 8, no. 3, p 6-19 (2004)

[32] Wischhof, Lars, André Ebner, and Hermann Rohling. "Information dissemination in self-organizing intervehicle networks." *Intelligent Transportation Systems, IEEE Transactions on* 6, no. 1 (2005)

[33] Camp, Tracy, Jeff Boleng, and Vanessa Davies. "A survey of mobility models for ad hoc network research." *Wireless communications and mobile computing* 2, no. 5 (2002)

[34] Bai, Fan, Narayanan Sadagopan, and Ahmed Helmy. "IMPORTANT: A framework to systematically analyze the Impact of Mobility on Performance of RouTing protocols for Adhoc NeTworks." In *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, vol. 2, pp. 825-835. IEEE, (2003)