UNIVERSITY OF BOTSWANA

MSc Computer Science Dissertation

A Greedy Next-hop Selection Module for Vehicular Ad-hoc Network Routing Using the A* Algorithm Method

by

Gbadebo Oladeji-Atanda

Submitted in partial fulfilment of the requirements of the Master's degree

in the Department of Computer Science

Supervisor

Prof F J Ogwu

December 2014

Contents

List of figures

List of tables

Acronyms

MANET – Mobile Ad-hoc Network

VANET – Vehicular Ad-hoc Network

WSN – Wireless Sensor Networks

V2V – Vehicle-to-Vehicle

ITS – Intelligent Transport System

DSRC – Direct Short Range Communication

WAVE – Wireless Access in Vehicular Environments

WSA – WAVE Service Adverts

IDM – Intelligent Driver Model

AODV – Ad-hoc On-Demand Vector Routing

GPSR – Greedy Perimeter Stateless Routing

GPS – Ground Positioning System

MFR – Most Forward within Radius

NFP – Nearest with Forward Progress

CR – Compass Routing

NC – Nearest Closest

A* – A-star algorithm

S* – Successive A* algorithm

DIST – Distance mechanism

Dedication

I dedicate all my works to the glory of God always. This work is further dedicated to the love of my wife Beatrice and children Simon, Abigail and Mary; all of who have had to endure almost three years of unintended separation from me during the period of completing this programme. Similarly, I dedicate this work to the comfort of several people who both they and I suffered inaccessibility to each other during the period 2012-2015. Finally, I dedicate the work to the honour of my late mother Eunice Mopelola, and my aged father Emmanuel Oladeji Atanda.

And further, by these, my son, be admonished:
of making many books there is no end;
and much study is a weariness of the flesh.
[98]

Let us hear the conclusion of the whole matter:
Fear God and keep his commandments:
for this is the whole duty of man.

Ecclesiastes 12:12, 13

Acknowledgements

I express appreciation for the guidance of Dr G Malema, who assessed my initial research interest and thoughts on sensor networks and directed me to Prof F J Ogwu, whose impetus for vehicular ad-hoc networks routing that I eventually chose to study invigorated me further. I express thanks to Prof Ogwu who became my supervisor, and generously financed my procurement of the EstiNet network simulator. Andrew Anam gave me a little push onto linux systems, while Bigani Sehurutshi bailed me out on occasions. Alex Moyo accommodated me cordially in the Network Lab of the UB Computer Science Dept. Largely, I enjoyed the supports of academic staff, technicians and colleagues in the department.

At the start of this programme of study, I was buoyant financially; but by the time of completion, I was virtually in rags. My gratitude goes to several people; God recognizes them all by name that liberally rendered to my necessities. I am thankful to the beloved family of Otu, namely Lawrence and Charity, together with their children Jude, Oge, Oyinye, Oluchi and Victory, who in simplicity and sincerity offered me a warm lodging in Gaborone during the one year protracted period that the dissertation lasted.

Statement of Originality

I, Gbadebo Oladeji-Atanda, declare that this dissertation is my own work. I have duly acknowledged and referenced the sources used in the work. This dissertation is original work not submitted to any other institution for the award of a degree or diploma.

………………………..	……………….
Signature	Date

Approval Page

Abstract

The potentials of traffic data sensing and vehicle-to-vehicle (V2V) communications for advancing traffic safety and efficiency at low costs is stirring the IEEE 802.11p vehicular ad-hoc network (VANET) type. However, the VANET milieu of incessantly high mobility nodes renders contemporary routing protocols ineffective in it. The development efforts of an appropriate protocol for VANET routing have narrowed towards the position-aware greedy scheme, which involves making next-hop forwarding choices of a neighbour that is geographically closer to destination continually. Forwarding techniques in MANET/VANET that use the geographic greedy scheme to make next-hop selections include MFR, NFP, CR, NC and Greedy.

We propose an alternative technique for packets forwarding in VANET. We designed the S* next-hop selection method, based on the A* path search algorithm. In addition, we developed a next-hop search-space limiting mechanism using the GLAR protocol's baseline and DIST (distance) concepts. We utilised the EstiNet simulator to model and evaluate our designs in terms of unicast outgoing and incoming packets delivery success rates.

The performance graphs remarkably show the S* technique demonstrating better results than the common Greedy technique in VANET routing. We further realized a phenomenon of either the S* or the Greedy method functioning best during certain periods of the packets forwarding duration. These alternating efficacies suggest prospects for the development of some *hybridized-greedy technique* for a more optimal next-hop selections method in VANET routing. The results also demonstrated the DIST mechanism's impact on packets delivery rates. Besides, we simulated and show the inefficiency of table-driven routing methods in VANET as compared to the dynamic greedy and geographic forwarding type.

1.0 INTRODUCTION

1.1 Background

Vehicular ad-hoc networks (VANET) development has advanced progressively, over the past decade, from the background of Wireless Sensor Networks (WSN) and Mobile Ad-hoc Networks (MANET) to become another stride in ad-hoc networks technology [1][2][3]. These wireless network types consists of functional autonomous system units; which when deployed in a site have the capacity to self-organise, interconnect and interoperate without a central controller or a pre-existing infrastructure (Fig. 1.1).

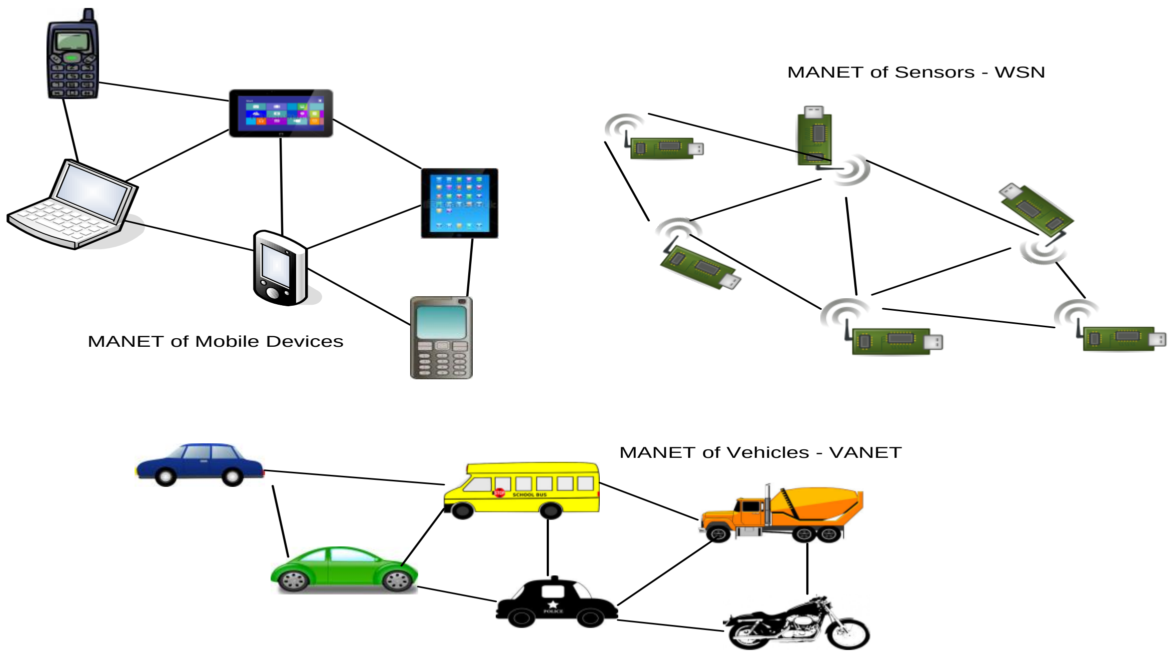


Figure 1.1: Types of mobile ad-hoc networks

An operational WSN deployment typically can contain several hundreds or thousands of cheaply produced nodes that could be positioned for sensing and harvesting some physical environmental conditions data; which are then periodically transmitted over the multi-hop ad-hoc network formed by the nodes to some base station. The idea of leveraging ad-hoc network technology therefore to the road arena for the sensing, acquisition and dissemination of traffic data in supporting the Intelligent Transport System (ITS) management forms the prime premise for the current VANET development efforts. VANET inquiry has included several aspects such as electronics, software engineering, automotive, transportation, security, and QoS considerations; however, the routing protocol development part has received much more attention from the research community [4]. Consumer services expected through VANET include automated traffic controlling, road safety and efficiency information dissemination, Internet access, data streaming, advertisements etc. [5][6][7].

The vehicular ad-hoc network (VANET) type consists of mobile vehicles that are equipped to communicate wirelessly with each other and with stationary Roadside Station Units (RSU). The RSU or Access Points (AP) could serve for service links, such as in automated toll collection point, or as a gateway to some WAN. The IEEE 802.11n and 802.11p (Fig. 1.2) interface standards in the internetwork protocols stack have been specified to aid vehicle-to-vehicle (V2V) and vehicle-to-roadside infrastructure (V2I) communication in VANET; while the US FCC, among others, has assigned to it the Dedicated Short Range Communications (DSRC) spectrum of 75 MHz at 5.95GHz [8][9].

Moreover, some major VANET experimental projects have been successfully implemented; which include the Intelligent Vehicle Initiative (IVI 1998-2004, USA), Vehicle Infrastructure Integration (VII 2004-2010, USA), FleetNet (2000–2003, EU), Cooperative Vehicles and Infrastructure Systems (CVIS 2006–2010, EU), and the Advanced Safety Vehicle programmes (ASV 1996–2000, 2001–2005, 2005–2007, Japan). The broad purposes for these field trials include the evolvement of VANET operating standards as well as the validation of its fundamental safety applications, e.g. the automated proximity-sensing and braking system.

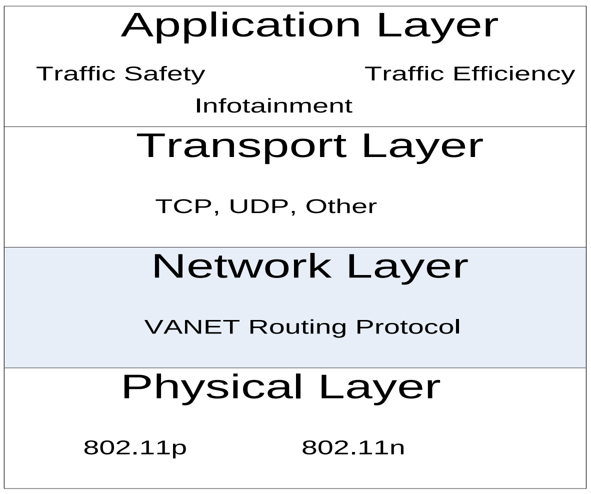


Figure 1.2: Vehicular ad-hoc networks protocols stack (adapted from [10])

So far, there is no standard packet routing protocol found suitable for VANET. Traditional network routing protocols, such as OSPF and RIP have been found unsuitable; neither are Ad-hoc On-Demand Distance Vector (AODV) [11] and Dynamic Source Routing (DSR) [12] protocols that are popular WSN/MANET systems. The unsuitability of these traditional routing protocols in VANET is primarily due to the high rate of mobility that is inherent in vehicular nodes, and the consequent rapidly changing network topology.

Other problems related to VANET operation include casualness with which nodes enter and exit from the set-up, its unbounded network size, and its high potential for frequent linkage partitioning. Nevertheless, VANET, unlike WSN, has little or no power supply and memory storage problems; since its power-supply is obtainable from the relevant automotive power system, and its memory size is easily scaled in vehicular hardware.

The following summarise general observations concerning the VANET routing protocol development efforts:

- No perfect routing protocol to use so far [13]
- Most designs handle only a particular portion of the problem [14]

- The solution trend is toward geographic greedy forwarding schemes [15][16][14]
- There are substantial variations in the way that protocol designs choose next-hop [17]
- It is hard for a single protocol design to handle the challenges of such a dynamic network, hence a requirement for some hybrid method [2].

Fig.1.3 is a timeline depiction of the geographic greedy protocol designs for VANET routing; although there are others that should have fitted into the figure, such as CAR [18], GLAR [19], and BAHG [20]. In general, these algorithms that employ greedy path search and forwarding method have appeared to be more viable in VANET than those requiring maintenance of routing tables.

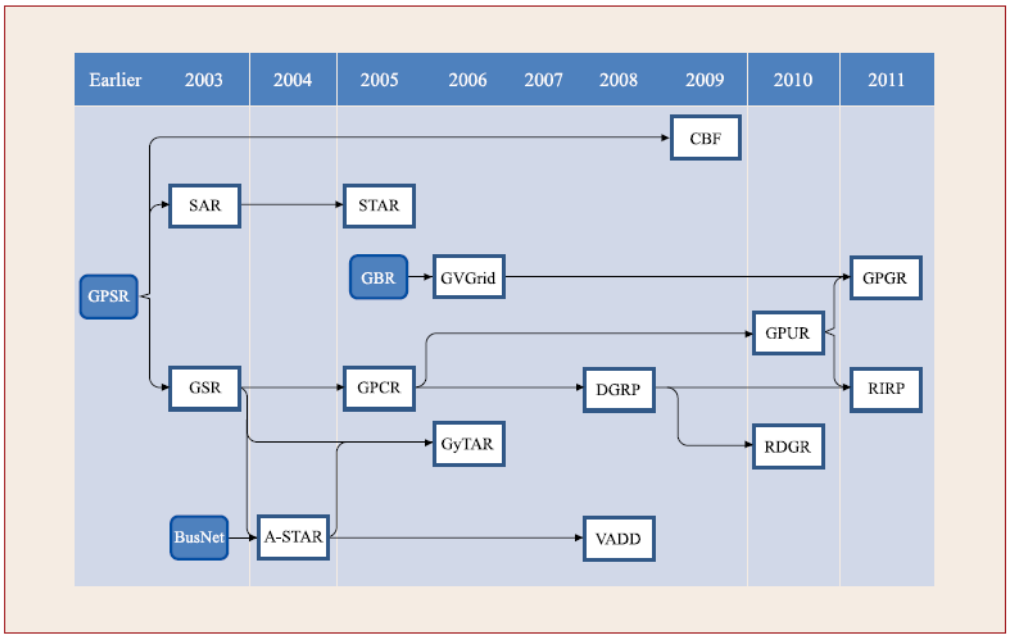


Figure 1.3: A development timeline of greedy routing protocols for VANET [15]

The Greedy Location-Aided Routing (GLAR) [19] protocol is of particular interest to our study on routing protocol designs due to its application of the baseline concept, which it uses to aid the geographic location-based routing path discovery. Note that the A-STAR (Anchor-based Street and Traffic Aware Routing) protocol listed in Fig. 1.3 is not attributable with the A* algorithm approach, nevertheless it utilizes the basic Dijkstra's greedy algorithm for routing. However, the current study also has interest in the A* algorithm as an alternative basis for implementing geographic greedy forwarding in VANET.

1.2 Related Work

The trend in VANET routing protocol designs has been in favour of location-aware and geographic greedy forwarding methods [15][16][14]; which we discuss in this section, and elaborate more in section 2.0. Similarly, we shall remark on the A* path search method that provides the algorithmic basis for our routing protocol module development in this study.

1.2.1 Location-aware greedy protocols

The purpose in VANET location-aware greedy routing protocol designs is to find suitable source-destination path linkages for effecting communication an ad-hoc topology, which itself floats over an underlying road network structure. Dijkstra's algorithm has commonly been employed in these greedy designs; for both charting routes over mobile vehicular nodes as well as over static landmarks such as junctions. The advancing of packets in location-aware greedy methods involves choosing next-hop that is geographically closest to the destination at each stage (Fig. 1.4). Hence, each node need only acquire or maintain network information about the positions of its immediate neighbours to be able to make forwarding decisions. The relevant information about the geographical location of nodes over an area's map or over the earth's topographical coordinates is obtainable from GPS units that are at present being installed in automobiles. Some survey of the greedy based routing protocol designs available so far have been conducted [15][16]; a sampling of which we proceed to discuss in relation to their multi-hop forwarding techniques.

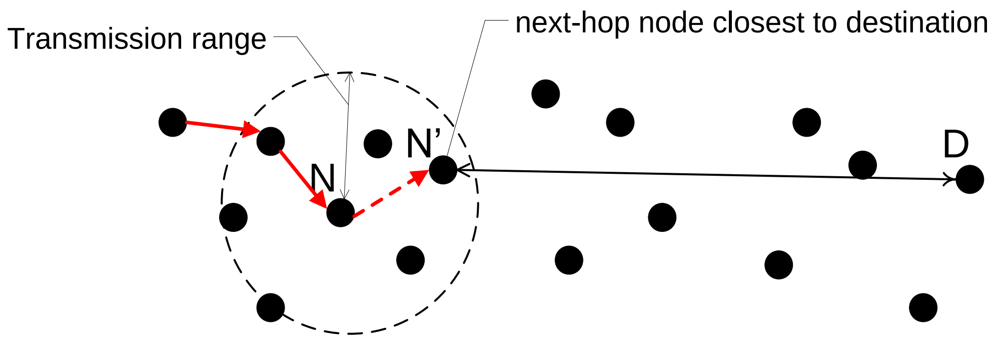


Figure 1.4: Location-based greedy forwarding

The Greedy Perimeter Stateless Routing (GPSR) [21] developer provided a seminal work in the use of geographic location for forwarding packets in mobile networks. The nodes in GPSR acquire and maintain neighbour location information through a periodic beaconing service. Every forwarding node makes locally optimal decision for each packet by choosing for it the next-hop neighbour node that is geographically closest to its header-marked destination. GPSR additionally provides a supplementary perimeter routing component for situations where no neighbour is geometrically closer to the destination than the forwarding node itself, i.e. when greedy forwarding fails at some local maxima. The problem of local maxima or routing void (Fig. 1.5) is a vital open issue in VANET greedy forwarding research [22][18].

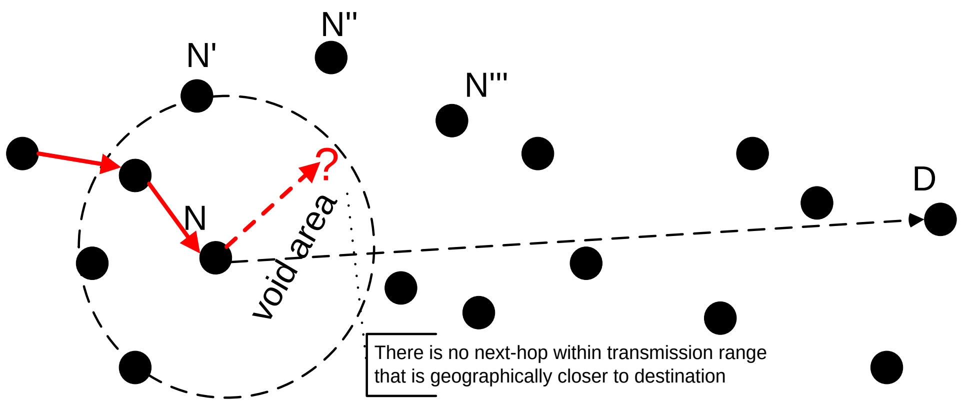

N''
N'
N'''
D
N
?
void area
There is no next-hop within transmission range that is geographically closer to destination

Figure 1.5: Void area problem in greedy forwarding

As shown in Fig. 1.5, the GPSR perimeter routing component has the capability to route round the void area by forwarding through N → N′ → N″ using the planarization technique and the right-hand rule for traversing a graph (if such a path exists). Then GPSR recovers to normal greedy mode after crossing the void with onward forwarding through N‴ to the destination D. However, the perimeter forwarding also fails if there is a lack of a face traversal graph setup around the void. GPSR had become popular as a VANET routing protocol benchmark; nonetheless, a flurry of protocol designs have come up after it because the solution it provides is inadequate.

The Greedy Location-Aided Routing (GLAR) [19] protocol is designed to improve on the Location-Aided Routing (LAR) [23] protocol, both of which are primarily designed for route discovery in MANET. These protocols in their operation initially at the source node do establish the destination node's position as obtainable from a GPS system; and then compute a geographic route map to the destination using the Greedy algorithm upon intermediate nodes' locations. LAR aims to reduce the number of nodes that respond to flooding-based RREQ messages, by restricting request to an area as depicted by the rectangle in Fig. 1.6. GLAR improves upon the search area reduction by employing a *baseline* value that is a source to destination virtual straight-line connection, surrounding which a greedy algorithm forwards a unicast RREQ message. The GLAR algorithm chooses the next-hop at each stage based on the option of being closest to the baseline but advanced farther away from the source node than the currently forwarding node. The GLAR protocol achieved improvement on connection lifetime and packets delivery rate with reduced control overhead as compared to LAR.

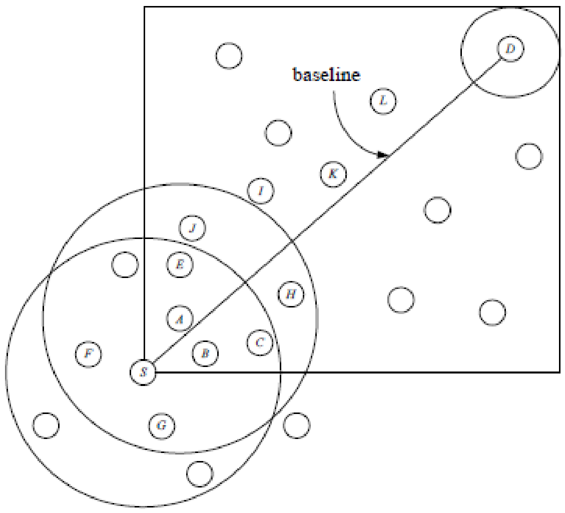


Figure 1.6: Route discovery baseline in the GLAR scheme [19]

The Hybrid Location-based Ad-hoc Routing (HLAR) [17] is not connected to the preceding, i.e. GLAR, but it is a hybrid method that profoundly combines principles of topology-based reactive routing such as is found in Ad-hoc On-Demand Distance Vector (AODV) [11] with that of geographic routing method, such as in Location-Aided Routing (LAR) [23]. The HLAR authors emphasize the view that no one type of protocol design approach can satisfy VANET protocols' complex requirements.

Geographic Source Routing (GSR) [24][15] is an early VANET position-based routing protocol design that uses static road topology map as a basis and supplement to geographic forwarding;

by the charting of routing paths along streets. The routing plan engages the Greedy algorithm to pre-compute the sequence of road junctions that packets have to traverse from source to destination; and the resulting route map is inserted in packets' headers accordingly. But GSR suffers from excessive packet overhead, as well as packets dropping along streets where traffic might be sparse [15].

Anchor-based Street and Traffic Aware Routing (A-STAR) [25] has a design that is similar to that of GSR, only it names junctions as 'anchor' points. A-STAR makes an improvement on GSR by providing packets recovery mechanism for sparse streets where voids might occur, i.e. situations of no next-hop node closer to destination. Recovered packets are then dynamically re-routed along some momentarily computed anchor path. Although A-STAR delivers 40% more packets than GSR and GPSR, it nevertheless also suffers from excessive packets overhead as GSR.

Greedy Traffic Aware Routing (GyTAR) [26] is similar to A-STAR in design; but rather than pre-compute the source-destination route, the next-hop path is dynamically determined by the Greedy algorithm at each forwarding junction or node as the case may be. In-between any two junctions, an 'improved' greedy method is supplemented by adjacent vehicular nodes' speeds and directions' information is used to forward packets. GyTAR incurs high computation and communication overhead [20].

Vehicle-Assisted Data Delivery (VADD)[27][28] routing protocol is a geographic packet forwarding design with DTN (Delay Tolerant Network) features, which provides mechanisms for managing incidental delay in data delivery. DTN capability serves some applications best, such as email delivery. VADD employs the predicable nature of vehicle mobility along the road network routes to choose forwarding along most viable streets' paths containing sufficient connecting nodes. Nevertheless, when a void occurs, VADD data packets remain in the carrier's buffer until it encounters a viable next-hop and then effects transmission. As such, VADD's design goal addresses sparse network environments; however, our aim is to address routing effectiveness in connected network situations.

The general pattern in the above set of protocol designs aptly reflect the notion of geographic greedy forwarding; with each design also variedly supplemented by some other decision-making features such as the prevailing traffic density along streets. Strategies that are adopted for the hop-by-hop forwarding of packets in a network depend on the dynamics of the specified environment. Three main packet forwarding schemes are identifiable in geographical or position-based VANET routing protocol designs. These are *greedy forwarding*, *restricted directional flooding*, and *hierarchical* strategies [22][29]; which though are not mutually exclusive. Regarding greedy forwarding, the main techniques described for it in the MANET/VANET literature [30][22][31] are:

- MFR – Most Forward within Radius [Takagi and Kleinrock 1984]
- NFP – Nearest Forward Progress [Hou and Li 1986]
- Greedy [Finn 1987]
- CR – Compass Routing [Kranakis and Singh 1999]
- NC – Nearest Closer [Stojmenovic and Lin 2001]

Fig. 1.7 depicts the above-listed techniques, where *S* and *D* represent the source and destination nodes respectively; and the circle depicts the transmission range of the forwarding node, which is also *S* in this case. Other labelled nodes show the potential next-hop choice from *S* in accordance to the pertinent method.

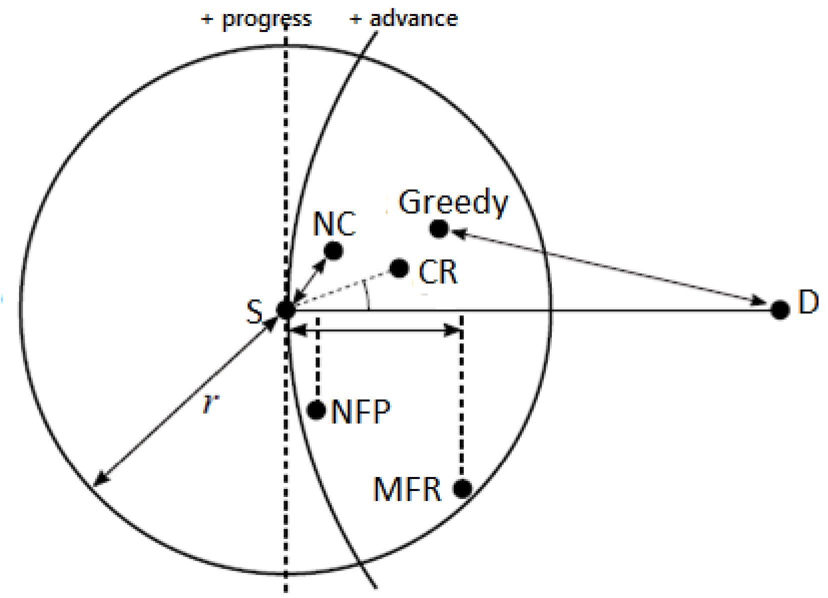


Figure 1.7: Variants of greedy forwarding [30]

In the current study, we show the design of another viable greedy technique that is based on A* path search technique rather than the basic Dijkstra's Greedy algorithm that has been generally employed by the routing protocol designs that were earlier described. Although A* is an extension of Dijkstra's algorithm, nevertheless it has attained a reputation as an efficient path search algorithm in several graph problems [32].

1.2.2 A* algorithm applications

The A* algorithm has been popularly used in optimal-path-finding problems over mapped terrains. The popularity of A* derives not only due to its efficiency in path finding, but also its adaptability and extensibility at different levels and in various domains of search problems [33][32][34]. It has been engaged in spheres that include robotics, networks, biology, database systems and games. For example, A* algorithm approach is found to outperform the generalized Dijkstra's algorithm in the time dependent shortest path problem in dynamic networks for the computation of fastest path routes that minimize travel time for truck drivers [35].

However, broadly viewed, A* algorithm's performance has been quite notable over static grids; whereas our extensive literature review did not show where A* algorithm's performance has been verified on shifting grids (or nodes), such as in VANET. But on the other hand, Dijkstra's greedy path search has been widely implemented in the VANET environment, as has been previously shown in this section. Hence the current study serves also as an opportunity to determine the performance of A* algorithm in environments of non-static grids.

1.3 Problem Definition

The need for some efficient routing protocol is currently a challenge to the emerging vehicular ad-hoc network (VANET) type. In particular, VANET requires an effective geographic greedy next-hop selection technique for routing packets over the characteristically high mobility and rapidly transforming VANET topology.

Problem Statement

There is no optimally efficient geographic greedy next-hop selection module in VANET routing protocol designs for the vehicle-to-vehicle (V2V) multi-hop packets relay.

Research Question

Can an optimally efficient location-aware greedy next-hop selection module for packet forwarding in VANET V2V communication be designed; with the use of A* path search method, which is further supplemented by a search space limiting function?

Supporting Questions

1 Can we design an optimally efficient next-hop selection routing module for VANET multi-hop data forwarding?

2 Can an efficient location-aware greedy packets forwarding technique be developed for V2V data relay in VANET?

3 Will A* algorithm path search approach for packets forwarding be effective and efficient in VANET where the grids are constantly shifting?

4 Rather than using only a static pre-computed baseline, as in GLAR [19], can baselines for enhancing limitation of search space be dynamically computed at successive forwarding nodes in VANET communication?

5 Can baselines' values be obtained for every potential forwarding node and used to serve as heuristics parameters at the currently forwarding node for the A* path determining algorithm?

1.4 Objectives of the study

In order to answer the questions above we state the related objectives as follows:

1) To design a geographic greedy forwarding next-hop selection technique for VANET packets routing using the A* algorithm method.

2) To design a supplementary search-space limiting function based on the orthogonal proximity of nodes to the virtual baseline that connects any forwarding node to the destination node, for reduced overhead in VANET routing.

3) To design a functional greedy routing protocol module for VANET packets forwarding, encompassing 1 and 2 above

1.5 Significance of the Study

This study thus makes the following contributions.

- Produce an alternative greedy forwarding technique that uses the A* algorithm method with capability for operation over VANET. Specifically, this study attempts to introduce another greedy forwarding technique to the menu of existing MANET/VANET techniques such as MFR (Most Forward within Radius).
- Produce an assessment basis for the A* algorithm's performance over terrains of shifting grids, which apparently has not been reported in the research literature.

1.6 Scope and limitations of the study

This study on routing protocol module design is concerned with vehicle-to-vehicle (V2V) inter-communication over VANET. Inter-vehicular communication could occur along the highway or within urban areas; this study attempts to include both. The study is not involving secondary issues, such as delay tolerance scheming in VANET, that are being addressed by some other research projects [2][36].

1.7 Approach

The process of the study involves ultimately producing a next-hop selection design that implement the A*-like greedy forwarding technique over dynamic grids of vehicular nodes, together with the associated supplementary search-space limiting component. The A* technique design is validated, through the use of a suitable VANET simulator, against the basic Greedy forwarding approach that is found in protocols such as are described in section 1.2.1. Similarly, the effectiveness of the search-space limiting function is evaluated by comparing some sample limits' performances graphs. We also use screen-shots from an animated simulation of the VANET scenario to evaluate the performance of position-aware greedy *hop-by-hop routing* of packets against the method that utilises pre-computed source-destination *route mapping*.

1.7.1 A* algorithm model

The A* algorithm uses the function *f(n) = g(n) + h(n)* to evaluate suitability of node links in path search problems [32]. The function g*(n)* is the weight of the distance from the source node to the current hop position node *n*, while *h(n)* is the weight of the distance estimate from the current node *n* to the goal node. In the VANET environment depicted by Fig. 1.8, *N* is the current packet forwarding node that has the option of choosing either *A* or *B* as the next-hop node, based on the A* evaluation function *f*.

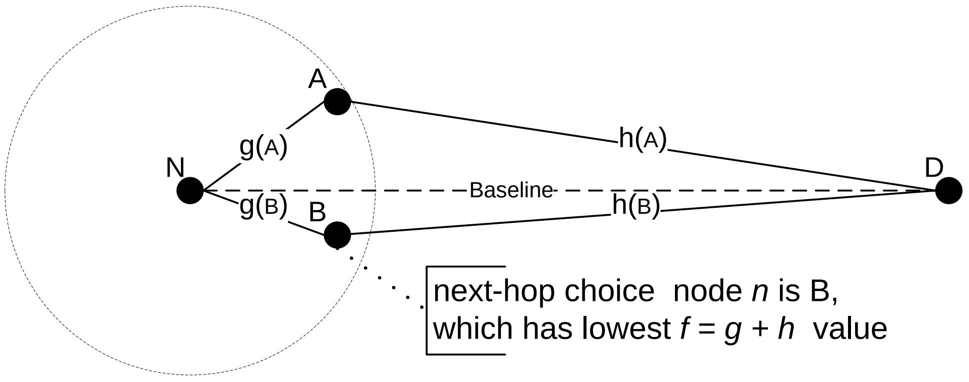

A
N
g(A)
g(B)
B
h(A)
Baseline
h(B)
D
next-hop choice node *n* is B, which has lowest *f* = *g* + *h* value

Figure 1.8: A* next-hop selection method

At any currently forwarding node, when the algorithm performs an evaluation of potential next-hops, the neighbour node that yields the lowest *f* value is to be chosen as the next-hop because it is deemed to lie along the least-cost path.

1.7.2 Basic S* next-hop selection model

In the S* implementation, an A* algorithm copy resides on each node in the network for deciding next-hop for each packet that arrives at the node; thus achieving multi-hop forwarding and dynamic routing in the VANET situation. Thus, we picture *successive A** instantiation at every succeeding forwarding node. Therefore for simplicity, we use the term *S* technique* to describe the VANET next-hop selection function that uses the A* algorithm method; and we refer to it as the *S* module* when combined with the path search space limiting supplement.

The GLAR [19] routing protocol method has been described in section 1.2.1. The peculiarity of the GLAR protocol model is its use of a pre-computed *baseline* to supplement routing path-search from source node *S* toward the destination node *D* as in Fig. 1.9. Going from *S*, the GLAR algorithm selects nodes *A* and *I* as next-hop respectively because they are closest to the destination along the geographic route to *D*; and are furthermore correspondingly closest to the baseline *SD*. However, where GLAR would choose node *K* as the next-hop in the subsequent stage, S* could choose node *M* based on the shift of *D* to *D'*. This is apparently a better choice in the dynamic environment of moving targets. The S* model extends the GLAR model without the ‘closest to baseline’ rule, but rather utilizes the baseline parameter as an heuristic input value *h(n)* in its A* algorithm type method. Furthermore, the S* module uses the baseline’s orthogonal limits to demarcate the extent of optimum next-hop choice bounds.

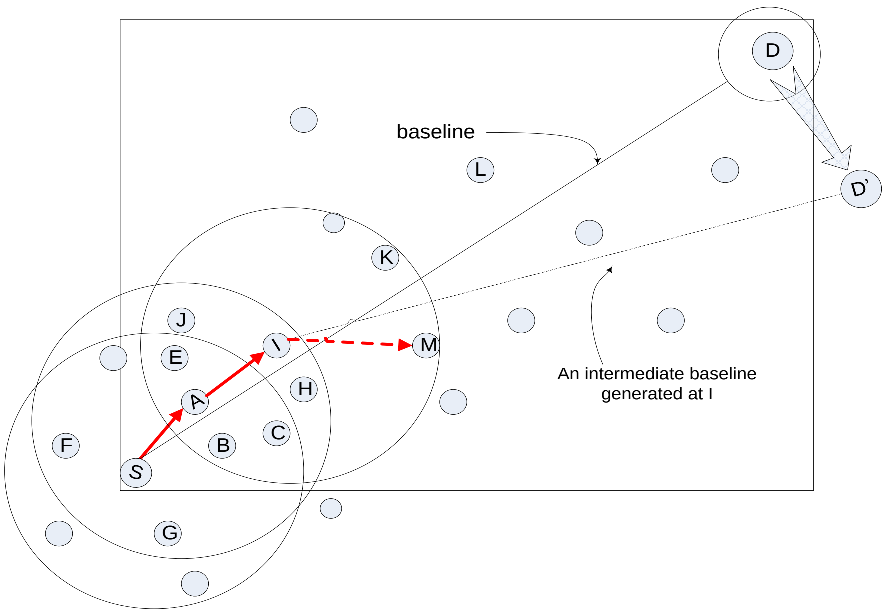


Figure 1.9: Basic S* routing model (adapted from [19])

We represent the differences between the GLAR technique and the S* as follows:

GLAR route discovery and forwarding scheme

- Greedy + closest to baseline hops choice, pre-computed entirely at source node

S* packets forwarding technique

- S* path computation at every hop using baseline values.

1.7.3 Basic algorithm functional calculations

The basic calculations required for baseline determination and destination node's displacement are as listed:

- Distances, (e.g. $f = g + h$ distances)

  For every distance PQ,

$$PQ = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \quad (1.1)$$

- The baseline

  For every baseline PQ, with points P(x_1, y_1) and Q((x_2, y_2),

$$(x_2 - x_1)(y - y_1) - (y_2 - y_1)(x - x_1) = 0 \quad (1.2)$$

  When the values of points P and Q are plugged into the equation (1, 2), it yields equation of a straight line to denote the particular baseline:

$$ax + by + c = 0. \quad (1.3)$$

- The orthogonal or vertical distance to baseline of a node as used in [19], and as shown in Fig. 1.10 is calculated as:

$$VDIST = \frac{ax+by+c}{\sqrt{a^2 + b^2}} \quad (1.4)$$

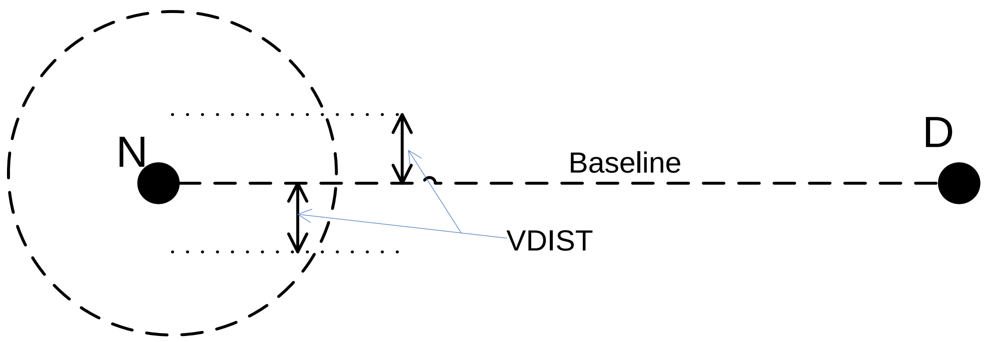


Figure 1.10: Vertical distance measures

1.7.4 Simulation

Experimental research effort in VANET has generally been by the use of simulators. VANET simulation had involved the coupling of some vehicular mobility generator such as SUMO with a data network simulator such NS-2; by importing the mobility trace file into network simulation environment [37][38][39]. However, this loose coupling approach does not support the manipulation of the vehicular objects' setting and behaviour differently from what a trace structure originally comprises; neither does it deliver an effective two-way affective feedback for the conjoined systems. Hence, VANET researchers have been demanding for composite simulator tools uniquely designed for VANET's environment modelling. It is further required that such integrated simulation facility should encompass realistic wireless channel modelling [39], as well as the comprehensive characterization of road topography and mobility at the macroscopic cum microscopic levels [40]. The nascent EstiNet simulator (aka NCTU-ns) appears to exhibit a good level of these qualities; and accordingly we employ its use in this research. According to [38], NCTU-ns exhibits a lot of advantages over traditional simulators such as NS-2 and OPNET, and it has been depicted as a most realistic VANET simulator compared to several others [4]; all of which we discuss in details in some latter sections.

1.8 Evaluation

We evaluated the efficiency of the greedy forwarding technique design and the overall performance of the S* forwarding module on packets delivery success rate as follows:

- Compare the end-to-end packets forwarding performance of the A*-like (S*) technique with that of the basic Greedy technique.

- Measure the effectiveness of the function that limits next-hop search-space to some set orthogonal regions around the baseline.

- Compare the performance of location-aware greedy forwarding that computes next-hop at every forwarding node with that of Ad-hoc On Demand Vector (AODV, [11]) reactive table-driven approach.

We compare performances of the methods principally on packets delivery success rate using graphed result charts. The originally expected outcome is an *at par* performance of the S* and basic Greedy techniques in greedy geographic forwarding; but with an added advantage of reduced packets travel distance in the use of the S* technique. Furthermore, the introduction of the baseline orthogonal limiting of next-hop space search component in the S* module is expected to promote reduced routing overhead.

1.9 Synopsis of the rest of the dissertation

The rest of the dissertation is organised as follows. In the next chapter a review of the VANET literature is given. Then chapter 3 describes related models and design features of the S* routing technique as well as the methodology of approach to its evaluation. In chapter 4 we present simulation results and analysis of the S* module implementation, while in chapter 5 we discuss the relevant findings and make conclusions.

2.0 LITERATURE REVIEW

This section provides a review of works related to the VANET routing protocol designs and development; particularly as concerns geographic greedy forwarding. We also discuss the A* algorithm approach which we employ in the formulation of the S* forwarding technique design for VANET routing. Finally, we describe the simulation environment for the evaluation of the design outcomes.

2.1 Routing Protocols in Conventional Networks

Computer networking development has matured into intricate levels of interconnected systems with very high degree of interoperability and scalability. Infrastructure or ad-hoc, wireline or wireless, are some of the main features that describe network tracts; which also have bearing on the type of routing protocol that may be applicable in each case. Network routing protocols are developed based on the need for multiple interconnected computing and peripheral devices to interact; which should happen automatically, seamlessly, and along optimally determined multi-link paths. Unlike in the fixed infrastructure networks, virtually all nodes in *ad-hoc* networks perform the function of routers in addition to being hosts. The VANET type of network in consideration is only conceivable in the ad-hoc and wireless mode. Specified routing protocols are the implement that enable the automated hopping of data packets from a source unit, across intermediate points, to some designated end.

2.1.1 Classic network routing protocol services

Classic network linkages feature dedicated traffic *router* devices, which have the capability to pre-learn path connections to every reachable destination within their respective domains; and can readily forward packets along chosen routes according to some policy, e.g. shortest-path forwarding requirement. Routing protocols' functions include route discovery, path maintenance and packets delivery across network grids by using some multi-hop forwarding technique. Routers in classic networks generally grow and maintain routing tables that portray connectivity to every part of their domain. The inherent advantage in table-driven systems is the immediate availability of routing information at the moment when there is a need to forward packets; thus eliminating *route acquisition latency* [41]. Network routing tables are normally constructed and updated periodically and dynamically too; however static configuration may be applied sometimes, such as in very small and static network. Whatever is the case, static routing configuration is not viable in VANET since vehicular nodes are in constant motion; and for the same reason the maintenance of routing tables is unrealistic.

Link-State (LS) and Distance-Vector (DV) are the two basic forms of classic routing protocols approach in infrastructure networks, which enable per-router broadcasts of connectivity information for the development of routing tables. In both the LS, where each node periodically broadcasts its local neighbours list, and the DV where each node broadcasts its global connectivity information, every other connected router uses these data to update their respective tables. RIP and OSPF are examples of routing protocols that respectively implement the above-mentioned routing approaches and have proved so effective in packets forwarding. Nevertheless, attempts to employ these protocols in VANET has been unsuccessful [42] due primarily to its topology dynamism.

2.1.2 Ad-hoc networks & VANET communication technologies

VANET belongs to the class of ad-hoc networks, which operate without fixed infrastructure; but the nodes are equipped with transceiver devices for enabling wireless communications. Other existing ad-hoc systems include military tactical networks, Personal Area Networks (PAN), Body Area Networks (BAN), home networks, disaster management networks, WSN, and MANET [43].

A range of IEEE interface standards have been defined for the ad-hoc network types [43][44]. The Wireless PAN (WPAN) IEEE 805.15.1 (aka Bluetooth) operates a short transmission range of 10m and 1mbps data rate at the globally unlicensed frequency band of 2.4 GHz. A Bluetooth supported network can contain at most eight mobile devices to form a piconet consisting of one designated master and n ($n \leq 7$) slaves, beyond which new piconets are hatched and scale into a scatternet cluster when the participating devices' number increases. The WPAN IEEE 805.15.4 (aka Zigbee) is defined for very low-rate (250kbps) data transmission requirements such as in WSN, wireless keyboard, and medical sensors. The WLAN IEEE 802.11 (WiFi) specification is for systems that require higher bit rates. IEEE 802.11b operates at the frequency band of 2.4 GHz and data rate of 11mbps, which is appropriate for wireless access in hotels, airports, etc. The IEEE 802.11a operates at 5GHz range and data rate of 54mbps for higher capacity support in environments such as offices. The IEEE 802.11n operates at both 2.4 and 5 GHz levels with data throughput of up to 600mbps and transmission range of 250m, which makes it suitable for use in campus settings.

IEEE 802.11n and 802.11p interface standards are specified for the VANET environment [8][4][9]. IEEE 802.11p specification is particularly for wireless access in vehicular environments (WAVE), in the US licensed ITS band of 5.9 GHz, with support for transmission range of up to 1000m and fast data rates of up to 27mbps. Both 802.11n and 802.11p are dedicated short-range communication (DSRC) standards for both vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) wireless connections as in Fig. 2.1. An exchange of information between users in separate vehicles or an automatic collision avoidance operating mechanism between adjacent vehicles would involve a V2V connection; while a V2I connection would be a link to a Road-Side Unit (RSU), for perhaps provisioning of connection to the Internet.

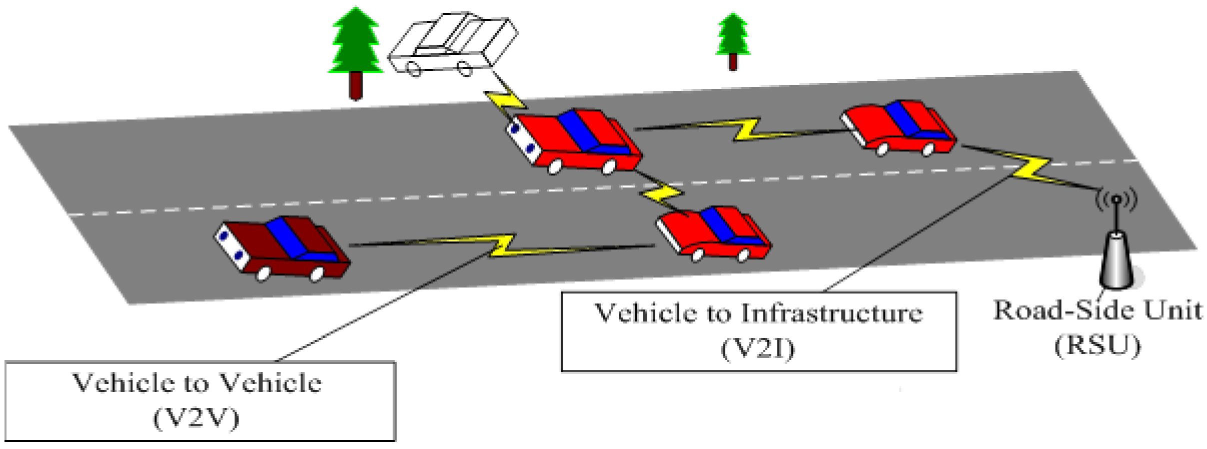


Figure 2.1: VANET communications structure

The general expectation is that the VANET system shall eliminate the existing cumbersome and costly ITS infrastructures sited on road networks. Considering the developing regions of the world where such traffic facilities are scarce or are easily vandalised, e.g. in most of sub-Saharan Africa, the VANET solution for transportation management offers a better prospect [45].

2.1.3 WSN routing schemes

The successfulness of Wireless Sensor Networks (WSN) applications in military and environmental monitoring tasks have been a significant influence on the upcoming idea of VANET [46][47]. A WSN node consists of integrated microcontroller, sensor probes, transceiver and a battery power unit. These nodes can be deployed deterministically or randomly in their environment of operation (Fig. 2.2); and could be few in number such as in home systems, or could be in thousands such as in forest monitoring distribution. WSN is typically a data-centric system, with all nodes primarily performing the function of sensing and gathering ambient physical data; hence, the requirement for IP-addressing type of procedures is unimportant in its topology configuration. However, locational awareness is vital at each node for the purposes of gathering data from the physical area of placement as well as for relaying data items along geographically determined next-hop neighbourliness route toward some specified base station or destination.

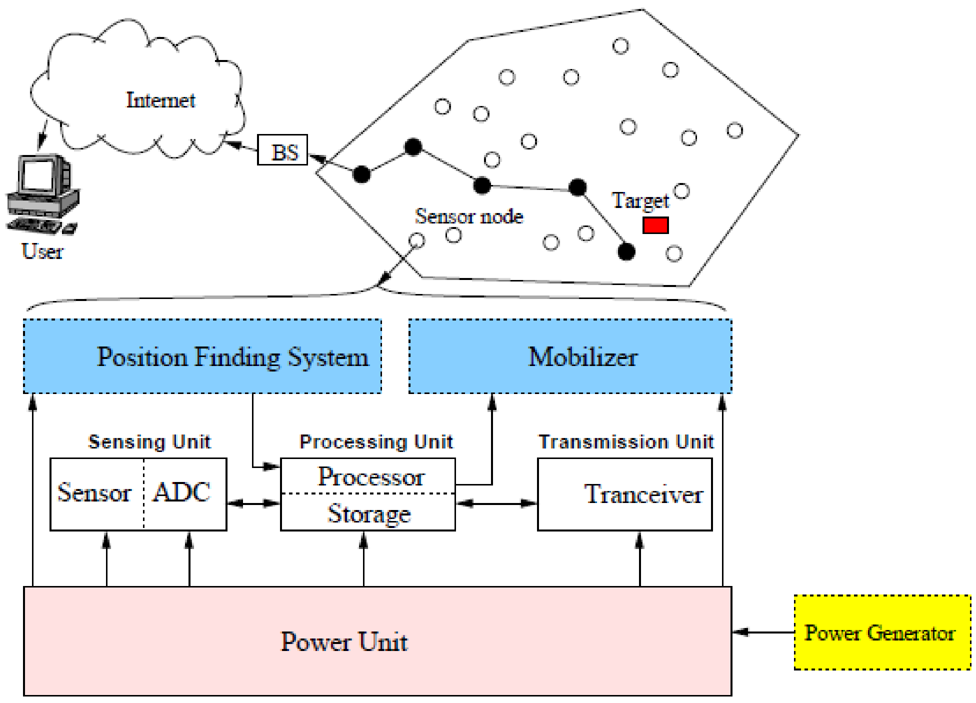


Figure 2.2: WSN configuration [46]

In military deployments, WSN provides utility such as the detection of troops' movements, identification of enemy positions, etc., while in environmental deployments, physical conditions such as temperature and pressure data can be continually sensed and logged. The possibility of having vehicles equipped with sensor units to harvest and disseminate road traffic data for safety and flow efficiency management is being envisioned as a better and cheaper way of doing business by the automotive sectors [6][48][49].

WSN routing protocol schemes involve using methods such as flooding and gossiping in data dissemination; as well as employing multi-hop data forwarding strategies that includes combinations of flat, hierarchical and location-based designs (Fig. 2.3). These routing schemes are respectively exemplified by some standard routing protocols, including SPIN, LEACH and GEAR that are described below. The effectiveness of multi-hop location-based routing in WSN deployments has thus influenced the adoption of this approach in VANET routing protocol designs [15][16][14].

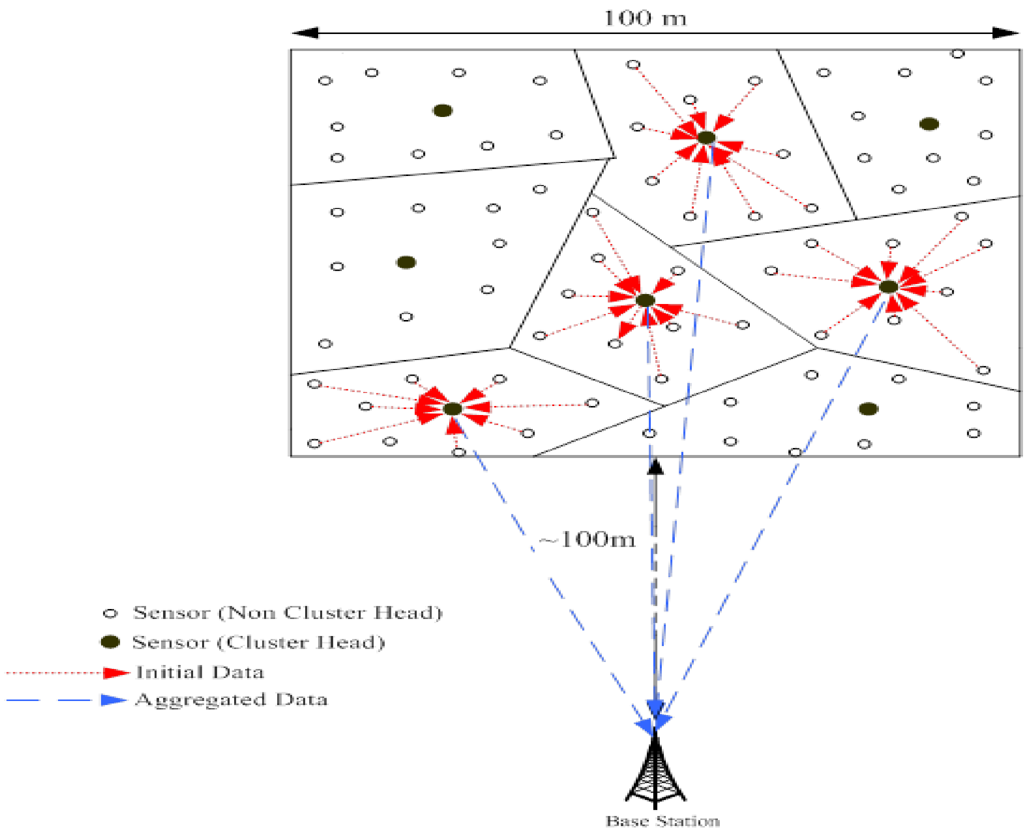
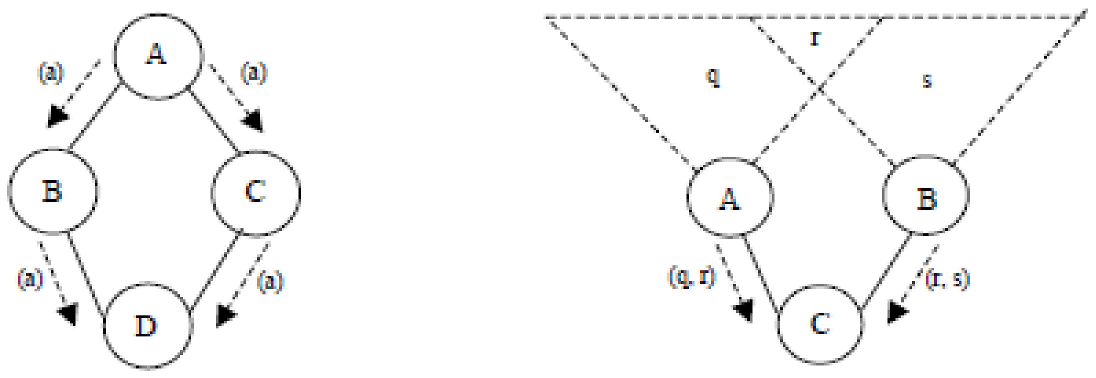


Figure 2.3: WSN hierarchical design [50]

2.1.3.1 Flooding and Gossiping - Data dissemination methods

Flooding is a basic method of transmitting or disseminating messages in a network by requiring every node receiving the message to also broadcast it to all its 1-hop neighbors irrespective of whether the neighbor had already received the same data or not; but only the concerned node(s) acknowledge such message while others may further broadcast or discard it. The drawbacks associated with flooding include the problems of *implosion* and *overlap* as depicted in Fig. 2.4. Moreover, flooding is fraught with *resource blindness*, i.e. uncontrolled power resource consumption [51], which is particularly undesirable for WSN nodes that have their lifespan dependent on their respective battery's lifetime.



(a) (b)

Figure 2.4 (a): The implosion problem – multiple sources (i.e. B and C) forward same data copies to a common one-hop neighbor (D), which is not necessary.

Figure 2.4 (b): Overlap problem– same data copies, covering an overlapping geographic zone (r), forwarded to neighbor (C) by multiple sources (i.e. A and B).

Figure 2.4: Flooding drawbacks [51]

The *gossiping* approach attempts to curtail implosion caused through simplistic flooding by making any forwarding node to transmit to only one randomly picked next-hop neighbour; but all nodes eventually receive the message. This method has the disadvantage of data propagation delay.

Flooding (and gossiping) message propagation method have been applied in some standard ad-hoc routing protocols in their route discovery schemes,including Ad-Hoc On-Demand Vector (AODV) and Dynamic Source Routing (DSR) protocols that we discuss in section 2.1.4; but these protocols are not efficient in VANET environments.

2.1.3.2 SPIN and DD – Flat WSN protocols

Sensor Protocols for Information via Negotiation (SPIN), an early data-centric routing design, is a family of flat protocols [52][51][46]. SPIN variants include SPIN-1, SPIN-2, SPIN-BC, SPIN-PP, SPIN-EC and SPIN-RL; with each type bearing some optimization according to some specified requirements such as energy conservation concern in SPIN-EC. The principal features in the SPIN protocols are (1) data negotiation among nodes before transmission to avoid duplications, and (2) nodes' allocation to routing tasks depending on their available energy levels. Nodes that have data to broadcast advertise such intent through a short ADV metadata message. If a recipient does not already have the advertised data then it makes a request with a REQ message, and subsequently receives the DATA message. Directed Diffusion (DD) [46] is also a flat WSN protocol, which however serves better in large networks where it guides data harvesting queries to only those nodes that are located around the phenomenon of interest. Moreover in DD, the data from multiple sources are consolidated en-route the reverse path (Fig. 2.5) to the query source [46]. Most user applications that are being developed for VANET traffic safety and efficiency management are essentially data-centric [3], and could benefit from the SPIN and DD methods.

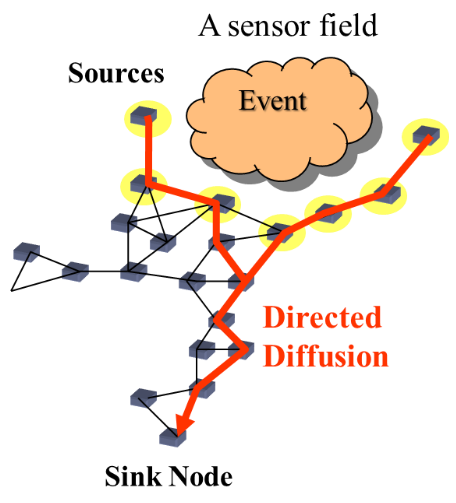


Figure 2.5: Directed Diffusion routing method [50]

2.1.3.3 LEACH – A Hierarchical WSN protocol

Low Energy Adaptive Clustering Hierarchy (LEACH) [53][46][52][51] protocol is a WSN hierarchical protocol type; where the network dynamically partitions into cluster regions of nodes for specified local data gathering tasks, and long range data transmission tasks are assigned to cluster heads. LEACH operates in rounds of two alternating phases: *set up* or cluster formation and *steady state* or data gathering. At the beginning of the set-up phase, a *k* number of random nodes elects themselves to be cluster heads using a probability algorithm that also supports rotational distribution of the role. The cluster heads then advertise themselves, so that each of the other free nodes can choose which cluster to join; and a node joins the one from where it receives the highest transmission signal supposing it to be the nearest. The LEACH steady state phase involves the aggregation and fusion of data by each cluster head for forwarding to the base station. Some VANET routing protocol designs utilize the cluster and hierarchical schemes, e.g. BROADCOMM and COIN, however much overhead is involved [54].

2.1.3.4 GEAR – A Location-based WSN protocol

Geographical and Energy Aware Routing (GEAR) [46][51] is a location-based routing protocol design for WSN. It uses geographical location and energy-content level information about nodes to make packets forwarding decisions. The idea is similar to that of Directed Diffusion described in section 2.1.3.2, that guides interests query to only the relevant region of the WSN area. The protocol uses heuristics that maintains some *closeness cost* (*learned* or *estimated*) at each node, based on the energy level and geographical distance of the node to any defined region *R*. A node that receives a packet for forwarding toward *R* makes a greedy selection of the next-hop based on the cost values. When within the target region boundary, the packet forwarding process can engage restricted flooding or recursive geographic forwarding methods. Greedy forwarding towards some geographic region is a method that has likewise been employed in some MANET routing protocol designs, e.g LAR [23]. Whereas GEAR uses additional information on available energy levels to supplement greedy forwarding choices, some VANET designs such as DREAM [55] use values including vehicle velocity and traffic density.

2.1.4 Basic MANET routing protocols

Interest and research in Mobile Ad-hoc Networks (MANET) of portable user devices (Fig. 2.6) predated the development of ad-hoc WSN; with routing protocols that were designed for the former already in place by the early 1980s, while the latter sprung up during the 2000s [41][56]. Initial MANET successful applications was in military tactical networks, for example in battlefield communications; but the increasing availability of enabling technologies, such as IEEE 802.11n and Bluetooth that are now being embedded in pervasive mobile devices is bringing about resurgence of interest in MANET [41][43]. Other areas of MANET applications include networking in disaster situations, where existing infrastructure might have been damaged [43].

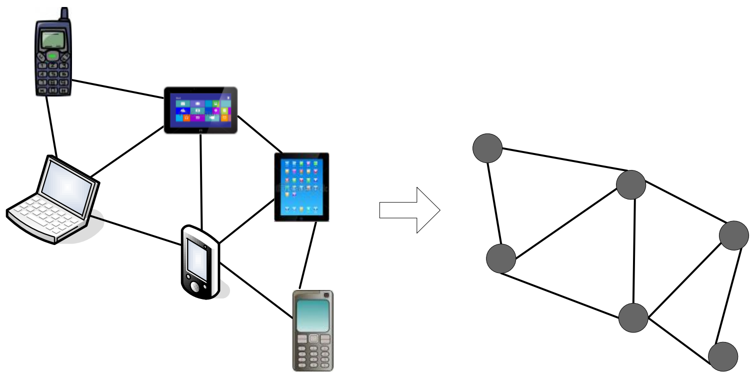


Figure 2.6: A MANET of mobile user devices

An important feature in MANET routing protocols' design is their location-awareness and support for mobile connectivity of nodes, albeit minimal, as devices can roam within some applicable transmission range. MANET has the following general characteristics and routing protocols' design goals [41][57].

MANET characteristics:

- Nodes density of 10-100
- Nodes mobility
- Varying network topology
- Limited bandwidth
- Limited power resource – as in battery operated devices

MANET routing protocols design goals:

- Geographic topology awareness
- Dynamic multi-hop packets forwarding capability
- Minimal routing overhead
- Loop prevention

In the following sections, we describe important features of some representative MANET protocols.

2.1.4.1 DSDV – A DV proactive MANET protocol

Destination-Sequenced Distance-Vector (DSDV) [58] protocol is an adaptation of the Distance Vector (DV) routing method that addresses the problems of looping and count-to-infinity to make it suitable for MANET through tagging each broadcast advertisement of routes' map with a sequence number so that receiving nodes may choose only the freshest entries [43][58]. DSDV was principally developed with mobile PCs in mind, where the number of participating nodes as well as their movements is moderate. Although DSDV is simple to implement, it generates very high control overhead, making it even unsuitable for VANET where frequent topology changes would accentuate the routing tables' maintenance overhead.

2.1.4.2 OLSR – An LS proactive MANET protocol

Optimised Link-State Routing (OLSR) [59] is an optimization of the Link-State (LS) routing method for reduced overhead in ad-hoc networks, through the introduction of Multi-Point Relays (MPRs). Each node in OLSR chooses a subset of nodes as MPRs from among its one-hop neighbours (Fig. 2.7). When a node broadcasts a message, only its MPRs can rebroadcast it to their respective next-hops, thus lessening flooding. Moreover, an LS broadcast of connectivity by a node has contents list only of the neighbours that have selected it as MPR, which also lessens message size [59][41][43]. OLSR works efficiently in dense networks, and it has been tested in VANET [60][61] where however constant topology changes makes it unsuitable as it affects the routing tables' maintenance [14].

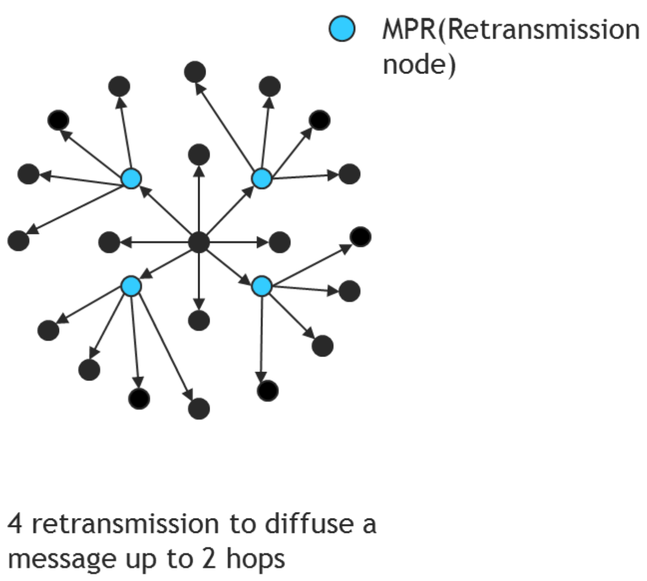


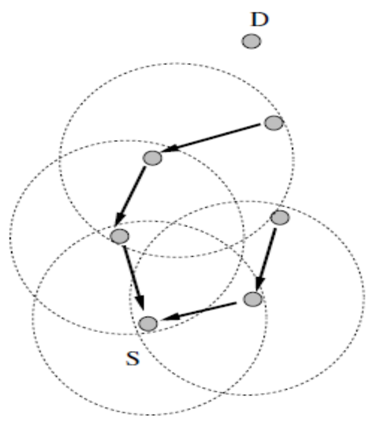
Figure 2.7: MPR scheme [59]

2.1.4.3 DSR – A reactive MANET protocol

Dynamic Source Routing (DSR) [12] is a reactive protocol where route discovery is performed *on-demand*, which although incurs some latency penalty. A source node broadcasts a RREQ that cites a destination if it does not already have relevant current routing map to that destination. The DSR route request message RREQ is flooded over the network from source *S*, and when a destination *D* or a node that has record of route to *D* receives the RREQ, it replies *S* with a RREP message containing route information from *S* to *D*. The RREP may be piggy-backed on an RREQ in the opposite direction, or the reverse path may just be defined as the routing path. The route map is subsequently embedded in packets by the source, and it is possible to assign varied paths to different data packets. DSR works well where nodes' movement is mild; otherwise its performance degrades fast, which makes it unsuitable for the VANET environment [62][63]. An important supplemental feature of DSR is *route caching* whereby every node listens in active promiscuous mode to network traffic so as to continually make updates of their routing tables accordingly, consequently minimizing future route discovery efforts.

2.1.4.4 AODV – A reactive MANET protocol

Ad-Hoc On-Demand Vector (AODV) routing protocol [11] is an extension of DSDV, using sequence numbering on its RREQ broadcasts as a mechanism to prevent looping. Each node in AODV maintains next-hop neighbour list as well as routing maps. A node that wants to send packets instantiates route discovery request if it does not already have the relevant map. In similarity to DSR's on-demand route discovery approach, AODV broadcasts RREQ while the corresponding RREP is unicast from the destination or some intermediate node that can provide route information that is current. However, packet headers in AODV do not hold route map; instead, reverse pointer links are set up for the routing path during the RREQ/RREP traversals (Fig. 2.8) through records kept by each hop. Tests of AODV in VANET show that it performs fairly well when conditions of node density and network size is moderate [14][54].



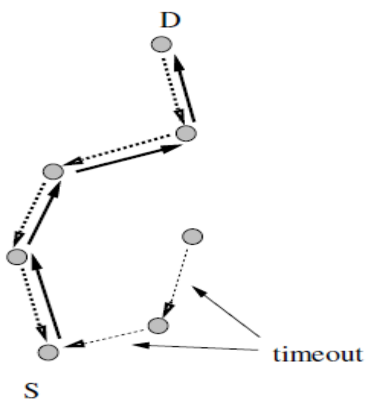


Figure 2.8a: AODV Reverse path

Figure 2.8b: AODV Forward path formation

Figure 2.8: AODV routing [11]

2.1.4.5 ZRP – A hybrid MANET protocol

Zone Routing Protocol (ZRP) [64] is a method that attempts to exploit the good features of both the proactive and reactive routing behaviours, using a hierarchical structure. In ZRP, a protocol such as proactive DSDV is utilised to perform *intra-zone routing* within a radius of *k* (*k*≥*1*) hops around every node that has to function at any time as source or forwarding node. When a destination is beyond a local zone, reactive route discovery method such as in DSR is engaged to perform *inter-zone routing* through *bordercasting*; which involves choosing the next-hop at the periphery of radius *k* of the forwarding node. Generally, structure-based protocols such as ZRP are scalable and bandwidth efficient; but they are inappropriate in VANET where the topological structure is unstable [45]. Furthermore for larger sized zones, such as is typical in VANET, the proactive routing part fails fast [65].

2.1.4.6 GPSR – A location-based MANET protocol

Greedy Perimeter Stateless Routing (GPSR) [21] is a protocol design where the nodes maintain location-awareness through the use of beaconing services, and the destination is determined using services such as GPS. Packets hold a target destination's co-ordinates information in their headers, which is used at intermediate hops to direct routing. A forwarding node determines the next-hop by choosing the neighbour that is geographically closest to the destination. If no next-hop neighbour is closer to the destination than a forwarding node, then a *void* is said to exist (Fig. 2.9); at which stage the algorithm switches temporarily to its *perimeter-forwarding* mode. The perimeter forwarding method involves dynamically constructing a planar graph of the nodes surrounding the void region, and uses the right-hand rule to effect a face traversal for packets forwarding around the edges of the void. In Fig. 2.9, node *x* is the point where packets go into the perimeter mode toward *D*, with routing winding through *xwvD*; after which greedy forwarding may be resumed if *D* is not the final destination. As of 2002, GPSR became a benchmark in VANET routing protocols' research [14][66]; whereas it was originally designed with general MANET situations in mind. This position of GPSR seems to be waning however, due to the evolvement of VANET routing designs that incorporate other essential protocol characteristics such as delay-tolerance and road network overlay prospects [14].

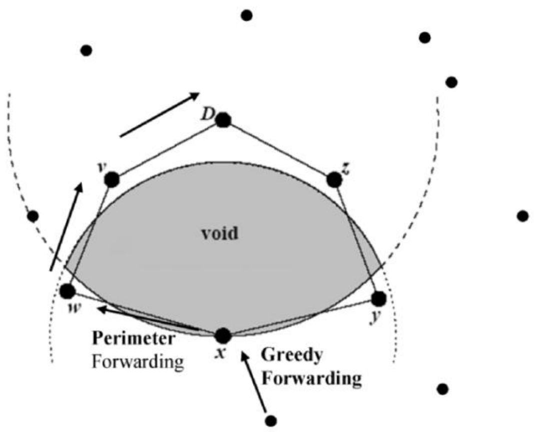


Figure 2.9: GPSR void repair strategy [21]

2.1.4.7 GPCR – A location-based VANET protocol

The Greedy Perimeter Coordinator Routing (GPCR) [67] protocol attempts to simplify GPSR's planarization and repair technique by the observation that in environments such as VANET, adjacent streets and junctions around a void area form a natural planar graph. Thus from a point where greedy forwarding fails, packets could be forwarded by routing along adjacent streets and junctions towards the final destination street. The depiction in Fig. 2.10 shows that a greedy forwarding void occurs in the separation between nodes *S* and *D*; consequently, the repair strategy performs the routing through *coordinator* nodes C_1 and C_2. GPCR exhibits increased packets delivery rate when compared to GPSR, however it does not address voids that are due to sparseness of nodes rather than street obstacles.

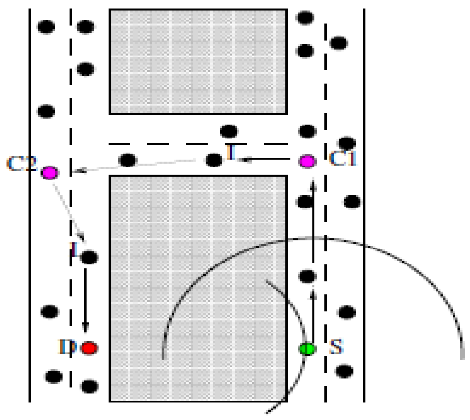


Figure 2.10: GPCR repair strategy [67]

The various MANET protocol types described above are mostly unsuitable for the rapidly transforming VANET topology type; consequently, researchers are looking for solutions that can satisfy the VANET add-on requirements.

2.2 Requirements issues in VANET routing protocol design

Generally, the goal of routing involves the effective transmission of data packets over a network of multiple intermediate nodes, through the use of methods that entail minimal delay and low overhead costs while achieving maximal throughput [54]. The VANET multi-hop routing protocol design therefore requires some particular considerations to be able to achieve the following general routing objectives [13].

- Low communication overhead

A low overhead in routing costs is desirable for achieving maximal data throughput. However, the costs heighten in VANET if the conventional routing procedures such as is in the proactive routing-tables' maintenance are implemented. Also note that the wireless media, that is generic in VANET, has a narrower bandwidth [57] in comparison to the wireline type.

- Low communication delay

Real-time systems in particular require very low communication delay. Low latency will be a requirement in core VANET vehicle-to-vehicle communications; for example, a system that has to communicate some imminent auto-crash event has hard delay constraints [54][4]. Hence, the adoption of the reactive and on-demand routing option that is inherently fraught with packets dispatch delay remains an integral issue in VANET.

The VANET environment is further characterised by the following effects [54]:

- High mobility rate of vehicular nodes

The rapidity of nodes' movements makes the link-lifetime of any communicating pair in VANET to last only a few seconds. For example, for some adjacent vehicles on a highway that are travelling by-passing each other at 100 km/h and with a radio transmitting range of 250m, the link between any pair can last at most 10 seconds [54][14]. Consequently, the applicable routing protocol designs must support lightweight algorithms with fast data transfer capacity.

- Rapidly changing topology and partitioning linkage

Continuous vehicular mobility leads to constant repositioning of nodes within the VANET environment, resulting in rapidly transforming topology as well as frequent linkage partitioning; therefore establishing a dedicated connectivity is practically infeasible in such an unstable condition. Furthermore, VANET is disposed to the arbitrary entry and exit of nodes, its unbounded network size, and its temporal cum district-level traffic density variations.

- Road structures and obstructions to wireless radio

Vehicular traffic runs on underlying road topography that consists of patterned highways, streets, intersections, traffic lights, etc.; all of which influence mobility. The aforementioned features are required in the faithful modeling of the VANET environment [14][40]. Furthermore in urban situations, tall buildings, trees and other structures such as bigger vehicles constitute interferences to wireless communication signals, and should be well characterized in the VANET data propagation modeling [13][14][54].

2.3 Directions in VANET routing protocol designs

Fig. 2.11 shows taxonomy of VANET routing protocols, which however is non-exhaustive; but it provides a general view of its designs, categorizations and trends. In this section we discuss the taxonomy as well as provide an overview on the emergent geographic forwarding approach.

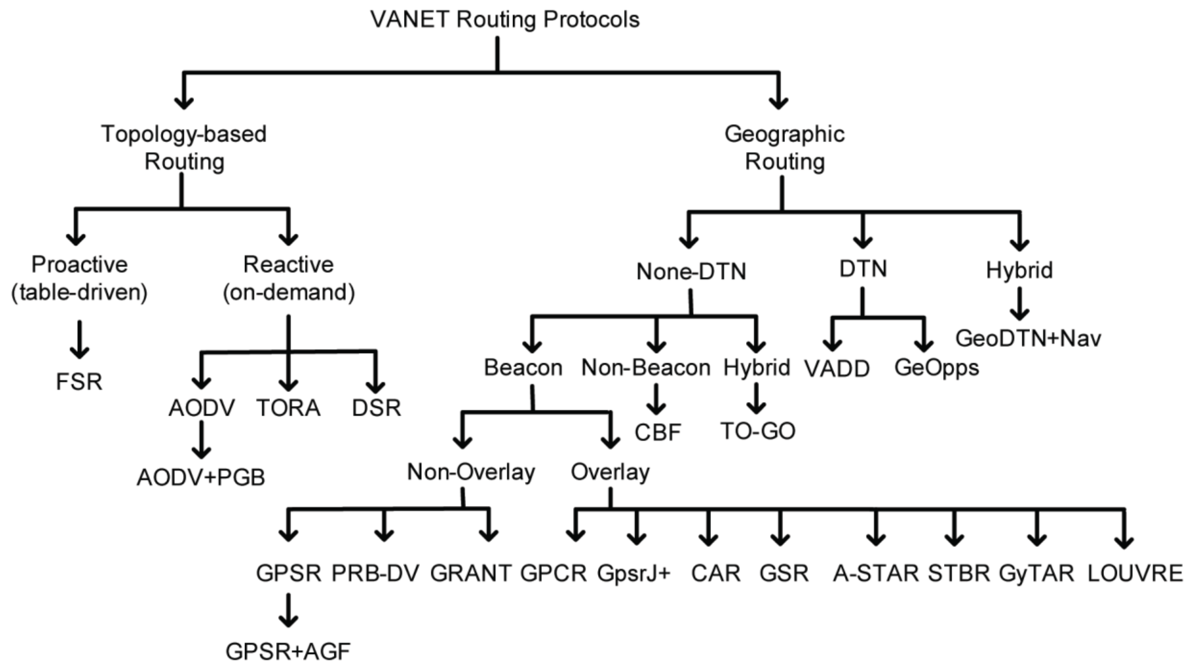


Figure 2.11: Taxonomy of VANET routing protocols **[14]**

2.3.1 Position-based vs Topology-based classification

The initial approaches to VANET routing protocol designs had involved applying adaptations of topology-based MANET protocols, which are classed as proactive or reactive. The nodes in proactive protocols maintain the map of an entire network segment in routing tables through periodic updates; and are most suitable where the topology rarely changes. Frequent topology changes, as in VANET, exacerbate tables updating and maintenance overhead. Reactive protocols on the other hand do not maintain routing tables as they perform route discovery only on demand; although this introduces some forwarding latency. In Fig. 2.11, the topology-based Ad-hoc On Demand Vector + Preferred Group Broadcast (AODV+PGB) and the position-based Greedy Perimeter Stateless Routing + Advanced Greedy Forwarding (GPSR+AGF) are adaptations for the VANET environment [68]. Basic AODV and GPSR have been described in section 1.1.4. Other types of AODV enhancements for VANET that are not shown in the figure include BAODV, AOMDV, and SD-AOMDV [65]. Despite these adaptations however, the topology-based ones have still been proved unsuitable for VANET; but the research community is honing on the geographic routing methods, which have demonstrated promising performances [69][70][22].

2.3.2 Delay-tolerance classification

The classification in Fig. 2.11 of Delay-Tolerant Network (DTN) and non-DTN types is within the location-based category. Delay-tolerance schemes allow packets to be buffered when necessary, especially during forwarding link cuts pending when connectivity is regained [71][2]]. The *data-muling* scheme further enables intentional vehicular-node carriage of buffered message for delivery to some next-hop or destination. Such delay-bounded designs may be with the aim of reducing bandwidth utilization when it will be tolerable to partly carry-and-forward data in non-realtime systems, e.g. in email deliveries [3]. Vehicle-Assisted Data

Delivery (VADD) and GeoDTN+Nav are examples of DTN designs; however, this area has not received much attention in VANET research.

2.3.3 Overlay vs non-overlay classification

The overlay characteristic of a routing protocol makes it to be aware of the underlying static road network topography, as it incorporates such secondary information to supplement location-based routing [14][69]. Overlay designs often engage forwarding packets along choice street corridors, which typically involves non-trivial decision making algorithms when at road intersections. In some cases Road-Side Units (RSU) backbone nodes are required to be installed at intersections to aid overlay implementation, e.g. in the Urban Multi-hop Broadcast (UMB) protocol [54]. But such RSU installations can quickly scale to thousands in cities that have numerous junctions; whereas an important aim for VANET advocacy is to eliminate the use of expensive and obtrusive roadside infrastructures. Also, routing along intersections in cities with short intervals between intersections may unnecessarily introduce delay in forwarding [20]. Nevertheless, VANET routing protocol designs are increasingly utilizing some forms of overlay information for forwarding strategy, particularly in urban environments.

2.3.4 Data dissemination classification

The sub-classification of VANET routing protocol designs showing unicast, multicast, geocast, and broadcast routing approaches are described in details in [65][13][2][72].

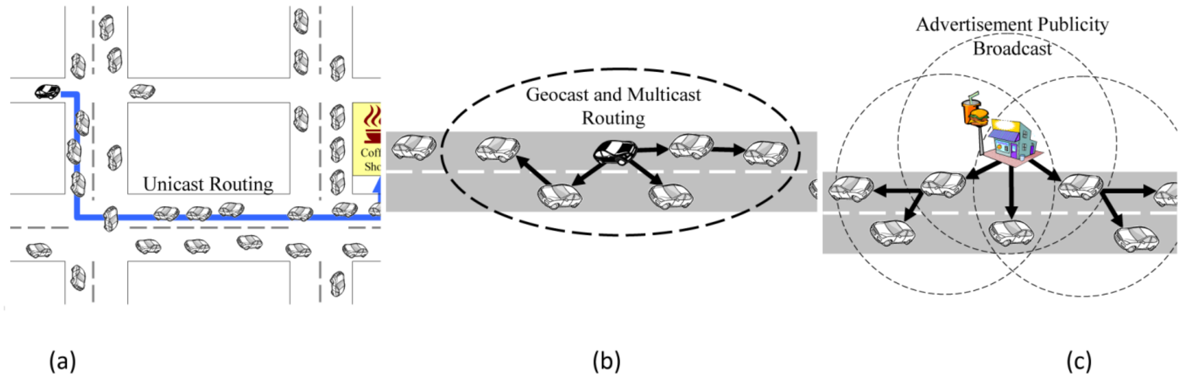

Unicast Routing
Cof
Sho
Geocast and Multicast Routing
Advertisement Publicity Broadcast
(a)
(b)
(c)

Figure 2.12: Data Dissemination Methods [72]

Fig. 2.12(a) depicts a *unicasting* of some query to a remote coffee shop. Most VANET routing protocol development efforts are unicast schemes where a source forwards packets to one specific destination [65], including the V2V communication links.

Multicasting involves disseminating data from a source to a set of interest nodes that may need same message copies. For example, information relating to roadblocks, accidents or other traffic conditions could be multicast to forewarn vehicles that are heading toward such incidents. Most multicast applications that are being designed for VANET require data dissemination within some specified region, i.e. *geocasting*, and is often implemented through directed flooding [54]. In Fig. 2.12(b), the broken down vehicle multicasts (or geocasts) a warning message to those vehicles currently traversing the incident zone. Traditional multicast routing schemes form trees or mesh map of multicast members within the network (to circumvent the flooding approach). VANET however does not support static routing structures, such as trees.

Broadcasting sometimes may be necessary for disseminating a message throughout an entire VANET domain, such as in Fig. 2.12(c) showing an advertisement broadcast. To avoid the broadcast storm problem however, efficient flooding designs for VANET have been proposed, including the use of only select nodes to re-broadcast [54][2].

2.3.5 Geographic forwarding strategies

Strategies that are adopted for the hop-by-hop forwarding of packets in a network depends on the dynamics of the environment. Three main packet forwarding strategies are identifiable for the geographical or position-based VANET routing protocol designs; which are *greedy forwarding*, *restricted directional flooding,* and *hierarchical* approaches [22][29]. In greedy forwarding methods, the neighbor closest to the target destination is chosen at each hopping stage; and this is popular in contemporary VANET routing protocol designs, which is a point of our focus in latter sections. Restricted directional flooding is similar to greedy forwarding, only that a forwarding node could select multiple next-hop neighbors to advance same data copy toward the destination. Hierarchical approaches involve formation of clusters, whereby only some elected heads take part in top-level forwarding, often with the aim of minimizing hop counts that packets have to travel. There are other forwarding strategies in use apart from the above-mentioned; for example the Mobility-Centric Data Dissemination for Vehicular Networks (MDDV) [73] that involves opportunistic forwarding, which however is not necessarily a geographic forwarding method.

2.3.6 Greedy failure recovery strategies

Greedy forwarding methods are quite suitable for handling forwarding in rapid mobility environments; more so because nodes need only maintain the local positional information about their immediate neighbours instead of keeping large routing tables [15]. However, greedy algorithms are short-sighted methods that make series of locally optimal choices with the belief to eventually achieve a globally optimum result; hence they are most ideal for problems exhibiting optimal substructures [74]. A local maxima failure occurs by the existence of some void area in an ad-hoc network, i.e. when no next-hop neighbour node exists that is closer to the destination than the forwarding node itself (Fig. 2.13). Local maxima frequently can occur in VANET because of network partition due to nodes' mobility, underlying road connectivity segmentation or obstructions to radio transmission in high-rise building areas. When a local maxima situation occurs, greedy method fails to continue executing; hence a recovery method is needed [14].

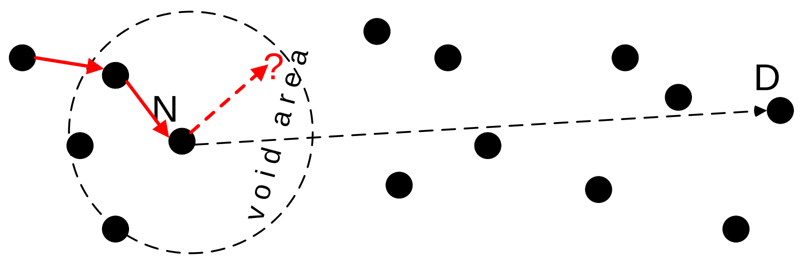


Figure 2.13: Greedy failure in the void area

The incidence of local maxima failure is yet an open problem in greedy routing research [22][18]. An innovative attempt at dealing with this void region problem is found in the design of Greedy Perimeter Stateless Routing (GPSR) protocol that dynamically constructs planar graph of adjacent nodes for routing around the void (see section 2.1.4.6). Nevertheless, planarization is only practicable when the relevant preconditions happen to hold, e.g. existence of some chance pertinent pre-positioned nodes. Other planarization routing designs are found in schemes such as Greedy-Face-Greedy (GFG) and Greedy Path Vector Face Routing (GPVFR) designs [30]. GEDIR [30] is a greedy method that handles void by simply sending packets back to the forwarding node to choose any other available alternative as next-hop; but excluding previously visited dead-end nodes. Detailed discussion on planarization and other techniques for routing around voids can be found in [30].

2.3.7 Packet swinging

The notion of *packet swinging* in greedy forwarding designs describes the situation where hopping packet units chase a moving target destination node around in the network [20]. Packet swinging is a design choice for the effective and accurate delivery of packets in the typically unsettled VANET environments. The routing path that a packet (swinging) takes is computed hop after hop to realize such effective forwarding in VANET [27] since any pre-computed path should soon alter due to the instability of node positions. Both Vehicle-Assisted Data Delivery (VADD) [27] and Greedy Traffic Aware Routing (GyTAR) [26] designs, shown in Fig 2.11 classifications, exhibit the hop-by-hop form of route computation. In GyTAR, every next-hop junction as well as every next-hop node engages in computations that select the next packets carrier. Its packet delivery ratio outperforms that of GSR and LAR that pre-computes their entire routing path before forwarding commences. The authors of GSR [24], which is discussed in section 2.4.1, also suggested the practice of computing next-hop link at each forwarding node as an alternative to stuffing packet header with routing map. We therefore conjecture that real dynamic approaches, such as packet swinging, are required for successful routing in VANET.

“Devising protocols for VANET may not be successfully accomplished by simple adaptation of protocols designed for wired networks and Mobile Ad hoc Networks” [75]. So far, the Internet Engineering Task Force (IETF) has not proposed any particular VANET routing protocol design as a de facto standard; however, some directions are emerging as location-based greedy forwarding approach is dominating the design forms.

2.4 Models of greedy location-aware routing protocols

Greedy forwarding is widely recommended as an efficient method for routing in VANET [22]; while its applicable computations for selecting next-hop options receives applaud for being simple and fast [76]. Several VANET routing designs use the Dijkstra’s greedy algorithm for forwarding, which has the policy of successively selecting the node closest to the destination at every hop. In this section, we expatiate on greedy routing protocols.

2.4.1 GSR - Geographic Source Routing

Geographic Source Routing (GSR) [24][15] is an early VANET position-based routing protocol design that uses static road topology map as a supplement for geographically forwarding packets along streets ‘where inherently there are no obstacles’, i.e. the circumventing of voids that may require perimeter routing as in GPSR. GSR routing plan engages Dijkstra’s algorithm to compute the sequence of road junctions that packets have to traverse from source to destination and the route map inserted in packets’ headers accordingly. The header-mapping approach has been criticised as bandwidth consuming, as it necessitates excessive overhead; moreover, packets dropping occur along those pre-computed streets paths with unforeseen sparse traffic [15]. Evaluation of GSR shows that it performs better than the native MANET topology-based protocols of AODV (Ad-hoc On-Demand Vector) and DSR (Dynamic Source Routing).

2.4.2 SAR - Spatially Aware Packet Routing

Spatially Aware Packet Routing (SAR) [77] protocol is a position-based and street-aware design with goals similar to GSR, i.e. a scheme to improve GPSR perimeter design by methods that rather proactively routes around holes or voids. SAR assumes global spatial awareness using GIS digital maps to identify areas of permanent local maxima or voids on the road topography.

In Fig. 2.14, an example of a permanent void or hole exists between terminal areas *C* and *E*. Considering the basic geographic forwarding scheme, if node *S* wishes to send packets to node *D*, it could choose node *A* as the next hop, which eventually leads to a local maxima at *C*; but the SAR routing scheme would foresee this static void area and rather choose *B* as the next-hop. SAR suffers the drawback of high packet overhead due to large header mapping.

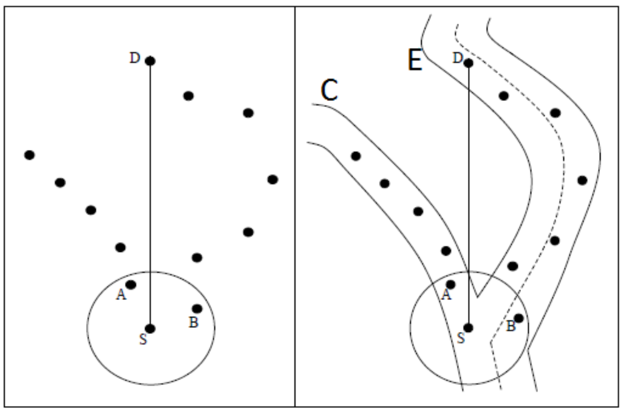


Figure 2.14: SAR's use of spatial awareness for geographic forwarding [77]

2.4.3 A-STAR - Anchor-based Street and Traffic Aware Routing

Anchor-based Street and Traffic Aware Routing (A-STAR) [25] is a position-based routing protocol that is designed for V2V communication in city environments. Its authors suggest that the basic greedy forwarding method is not suitable for city environments where there are high-rise buildings that pose as radio transmission obstacles; hence, in addition to location awareness, the protocol assumes street as well as traffic awareness.

In A-STAR, *statistically rated* maps describing regular city bus routes and *dynamically rated* maps of real time traffic information provide knowledge about streets with current vehicular connectivity. As like in GSR, A-STAR computes end-to-end routing path of sequence of junctions, called *anchors*; and it forwards packets greedily between anchor points. If it encounters a local maxima, the protocol's recovery strategy generates a temporal "out of service" message for that path, while it dynamically computes an alternative pathway for re-routing the packets. The salvaged packets would piggyback the out of service message for broadcast to the rest of the network to put such path temporarily out of use. A-STAR evaluation shows an accomplishment of 40% more packets delivery rate than GSR, and exhibits a better performance over GPSR. Its disadvantage lies in high packet overhead of junctions listing, which can quickly scale in city settings with numerous junctions.

2.4.4 VADD - Vehicle-Assisted Data Delivery

Vehicle-Assisted Data Delivery (VADD)[27][28] is a geographic packet forwarding scheme which incorporates the DTN (Delay Tolerant Network) feature. VADD uses the knowledge of predicable nature of vehicular mobility on roads, street topography, as well as prevailing traffic statistics, including nodes density and speeds; that are obtainable from installed on-board devices such as MapMechanics or Yahoo Smart View. The scheme evaluates and chooses streets' path that offer minimum packet delivery delay. Within the scheme, it implements the DTN carry and forward model with three modes of operation among which a packet carrier switches depending on its location: *straightway*, *intersection*, or *destination*. Along straightway stretch of streets it forwards packets using the greedy method, but when that fails, the carrier retains the packets until when in contact with an appropriate next-hop. Intersection decisions are quite involving, requiring non-trivial system of linear equations' calculations to select the most appropriate next-hop node that is heading in a promising direction.

In general, VADD outperforms Epidemic [78], which is also a DTN opportunistic forwarding routing protocol. It outperforms DSR and GPSR in packet delivery rate, delay rate, and overhead metrics in both sparse and relatively dense networks. There is the need for more study on sparse VANET environments, such as VADD addresses.

2.4.5 GyTAR - Greedy Traffic Aware Routing

Greedy Traffic Aware Routing (GyTAR) [26] aims to address routing in city environments, using street and junction-based forwarding, similar to A-STAR. GyTAR routing path is however dynamically determined at every forwarding junction and at every intermediate node. GyTAR assumes existence of location services that provide on-board information to each forwarding node regarding the current geographical position of the destination node, positions of neighbouring intersections, traffic density between intersections, distances between intersections, and speed plus direction of movement of neighbouring nodes. The neighbouring vehicles' speed and direction of movement data is to be gathered using hello beacon messages. In forwarding between intersections, an 'improved' greedy method that uses speeds and directions information chooses next-hop based on forecasted positions. A next-hop choice is determined at computation time t_1 to be the node that is predicted to be closest to the next intersection by the forwarding time t_2 ($t_2 > t_1$). The route from the current junction to the next is similarly determined on the fly based on distances and traffic densities, with the best way being one with dense traffic and next junction being the one that is geographically closest to the destination.

GyTAR exhibits significant performance improvement in terms of packets delivery ratio and end-to-end delay when compared to GSR and LAR [23]; but it is criticized for excessive computation and communication overhead [20].

2.4.6 GLAR - Greedy Location-Aided Routing

The Greedy Location-Aided Routing (GLAR) protocol [19] is an improvement on the LAR [23] design; both of which primarily implement route discovery and packets' header mapping in MANET. These protocols in their operation initially establish the locations of the source and destination nodes using GPS. They further acquire the destination node's velocity and time values for use in calculating search zone perimeters. The *request zone* (rectangular area) as shown in Fig. 2.15 isolates the potential next-hops' region, while the *expected zone* (circular area around *D*) defines where *D* may be reached even if it shifts position. The goal in LAR is to reduce overhead in route discovery by using the directed flooding method. GLAR improves LAR by employing the *baseline* mechanism, which is a computed virtual straight-line from the source to the destination; and the equation of the straight line is used to reference it. In Fig. 2.15, route discovery goes closely along the baseline that joins the source node *S* to the destination node *D*. The choice of each next-hop in the direction of the destination evaluates the greedy measures as well as the values of being closest in distance to the baseline. GLAR achieved better connection lifetime and packets delivery rate with reduced control overhead.

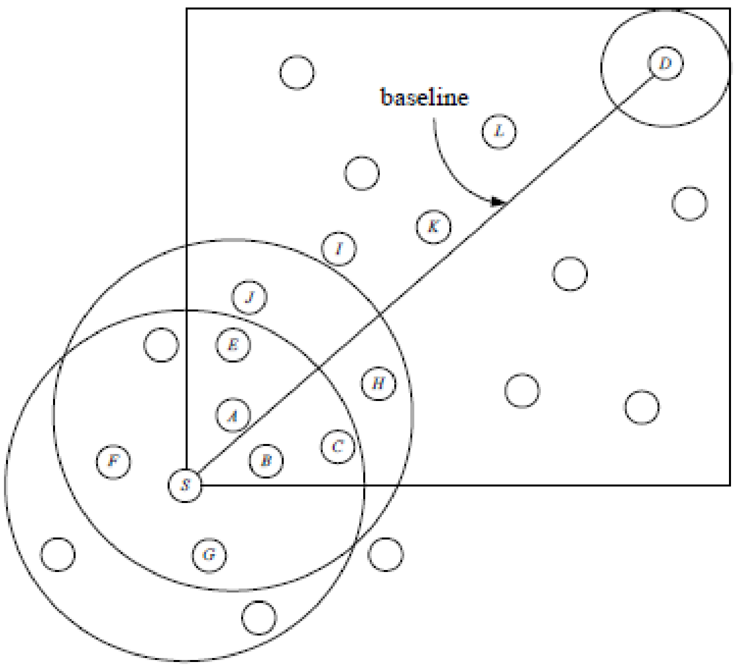


Figure 2.15: Route discovery baseline in the GLAR scheme **[19]**

In this study we use some of the GLAR protocol concepts within the A* algorithm method to design an alternative geographic packets forwarding module, which is comparable to the basic Greedy method that several VANET routing protocol designs have employed.

2.4.7 HLAR - Hybrid Location-based Ad-hoc Routing

Hybrid Location-based Ad-hoc Routing (HLAR) [17] is a routing protocol that is non-trivial in its complexity. The hybrid aspect of the protocol is that it combines principles of reactive routing with that of geographic routing. HLAR, among other purposes, aims to provide for a recovery mechanism when link breakage or void occurs during geographic greedy forwarding of packets. Normally an HLAR source node will use routing information that has been constructed in an AODV (Ad-hoc On-Demand Vector) [11] reactive manner to forward packets to the intended destination. However if route information becomes lacking at any forwarding node, it utilizes the geographic greedy method to continue packets' advancement. Moreover, if it encounters the greedy forwarding void, it engages the AODV's type of RREQ flooding in search of such neighbours that may have route information to the destination. According to its authors, HLAR is deployable in different kinds of VANET environments while it incurs lesser overhead when compared to other topology-based and location-based designs.

In this section, we have reviewed some position-based greedy forwarding routing protocol designs for VANET. There are protocol designs that explicitly provide mechanism for recovering from voids, while some pre-emptively sidetrack structural obstructions. In addition to the use of location information, most designs now use extra knowledge, such as street map and traffic density, to supplement geographic greedy forwarding decisions. In the next section, we focus on the greedy forwarding techniques used in MANET/VANET protocol designs.

2.5 Geographic greedy forwarding techniques in MANET/VANET

VANET position-based greedy forwarding designs that use geographic location information generally work under three basic assumptions [30][22], which are:

- Each node knows its own position's coordinates
- Each node is aware of its immediate neighbours' positions
- The destination position is known

The information over these assumptions may be provided by GPS, beacon messaging, or some other services such as cellular network detecting as proposed in [79]. The greedy forwarding techniques under discussion are applicable generally in MANET/VANET environments depending upon goals such as minimizing energy consumption or minimizing total distance travel to destinations. Figs. 2.16 and 2.17 depict these techniques with a routing plan from source node *S* to the destination node *D*. Node *S* may also be viewed as any forwarding node, with the circle as its transmission range. The roles of the other nodes are explained in the following subsections.

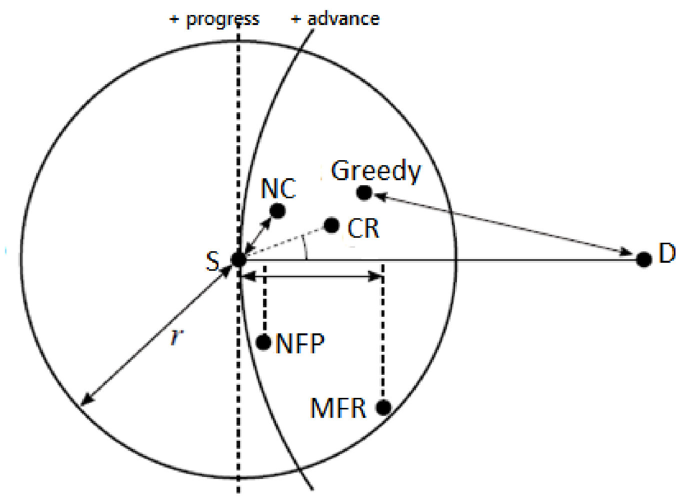


Figure 2.16: Variants of greedy forwarding **[30]**

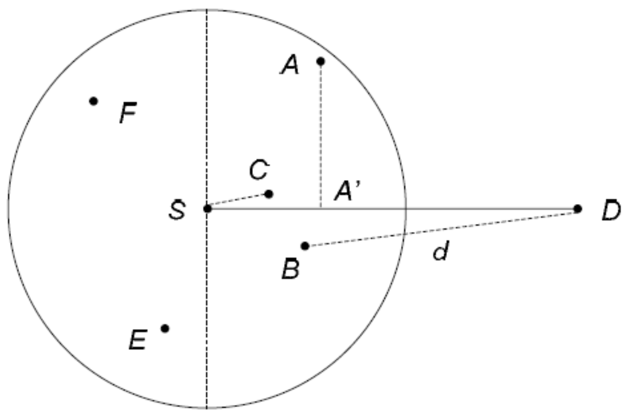


Figure 2.17: Greedy forwarding projections of MFR and NFP on the line *SD* **[31]**

2.5.1 Most Forward within Radius (MFR)

The Most Forward within Radius (MFR) [31] greedy technique chooses the node with the most forwarding *progress* towards the destination as the next-hop. MFR (Fig. 2.16) does not necessarily select next-hop with minimal distance to *D*, but achieves maximal forward progress from *S* towards *D* along the straight line *SD*. In Fig. 2.17, *A′* is a projection of node *A* unto line *SD*, and it shows farthest progress from *S* when compared to similar projection for nodes *C* and *B*. Benefits of the MFR method are that it minimizes the number of hops to be traversed towards *D* and is loop-free [22][31].

2.5.2 Nearest with Forward Progress (NFP)

The use of Nearest with Forward Progress (NFP) [31] greedy technique goes together with the condition of adjustable radio transmission range, and it aims at energy efficiency by choosing closest next-hop neighbor with *progress* towards *D* (Fig.2.16). This approach has high probability of minimizing collisions in the network [22][30]. In Fig. 2.17, *C* is the NFP node when projected unto line *SD*. NFP is mainly used in WSNs where energy management is an important issue. But [80] suggests an advantage of its use in networks with changing topologies, such as VANET, when concern is to reduce transmission collisions.

2.5.3 Greedy

The basic Greedy [31][30] forwarding technique chooses next-hop neighbour having least distance to the destination at each hop. Greedy (in Fig.2.16) is described as being quite suitable for large and dense network with frequent topology changes because its algorithm is simple and localized [31]; making it quite suitable for typical VANET settings. Greedy is also loop-free. In Fig. 2.18, Greedy and A* techniques' difference is described; where greedy next-hop choice from the forwarding node *S* would be node *B* because it is closest to the destination *D*; A* on the other hand would choose next-hop as node *A* because the distance *SAD* (i.e. *SA* + *AD*) is shorter than distance *SBD.*

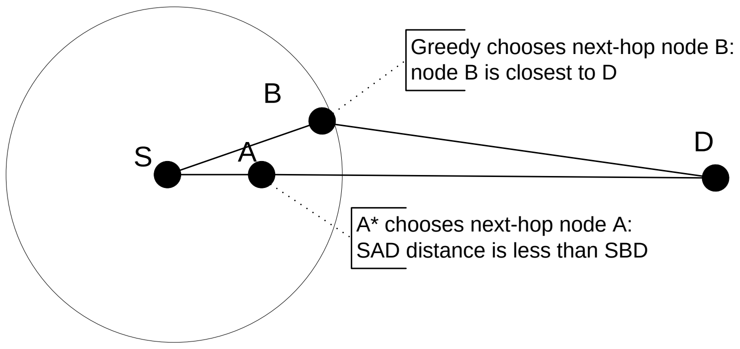

Greedy chooses next-hop node B:
node B is closest to D
B
S
A
D
A* chooses next-hop node A:
SAD distance is less than SBD

Figure 2.18: Comparison of Greedy and A* schemes

2.5.4 Nearest Closer (NC)

The Nearest Closer (NC) [30][31] technique chooses next-hop that is closest in distance to the forwarding node rather than to the destination node. Fig. 2.16 shows the NC node. NC addresses MANET power and energy cost savings concerns [81]; which although are regarded as trivial or no issues in VANET.

2.5.5 Compass Routing (CR)

The Compass Routing (CR) [30][31][82] technique aims at minimizing the total spatial distance a packet travels from source to destination, by choosing as next-hop those intermediate nodes that have minimum angular separation to line *SD* at each stage (Fig.2.16). In Fig. 2.17, ∠*CSD* is the minimum; hence, node *C* would be the CR's next-hop choice here. Fig. 2.19 compares CR to the GLAR [19] routing protocol's method that was described in section 2.4.6, which chooses next-hop greedily a node that is also closest to the baseline. Where CR would choose node *B*, the GLAR's choice would be node *A* since this is closest to line *SD*; notwithstanding the equality of ∠*ASD* and ∠*CSD*. Generally, CR ensures that packets routing goes closely in the direction of the destination; but not necessarily along the shortest path. CR is not loop-free [30].

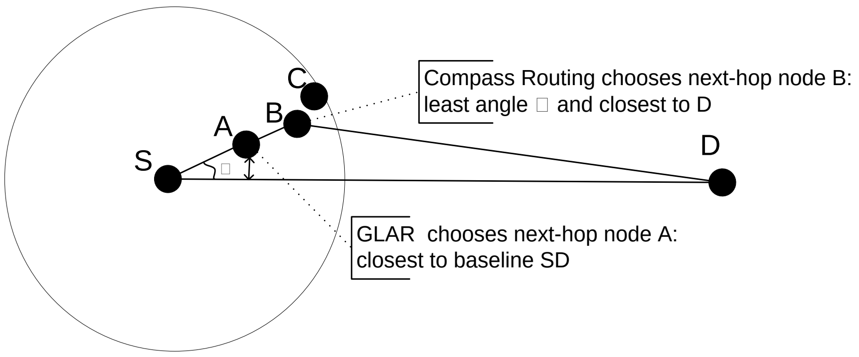
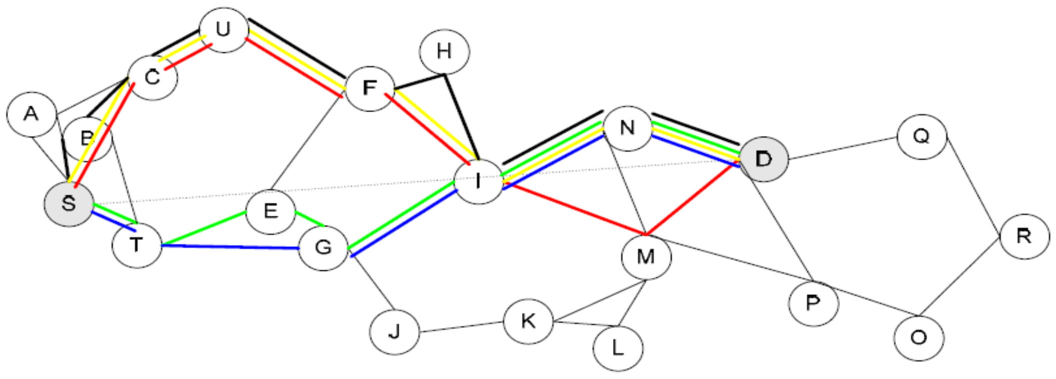

C
B
A
S
D
Compass Routing chooses next-hop node B: least angle and closest to D
GLAR chooses next-hop node A: closest to baseline SD

Figure 2.19: Comparison of Compass Routing and GLAR algorithms

2.6 A depiction of greedy forwarding techniques

The greedy forwarding techniques described in the previous section have been mostly developed for utility in ad-hoc networks and are applicable depending on the concerns being addressed in a network, such as the use of NFP to minimize transmission energy consumption. Fig. 2.20 describes the paths that some of the different greedy techniques would follow, going from *S* and *D*, in making next-hop selections in the given static graph [31]. The shortest path coincides with the A* algorithm path, for which it is renowned [32] as the choice method for finding optimally shortest travel distance.



Greedy: S-C-U-F-I-N-D
MFR: S-C-U-F-I-M-D
CR: S-T-E-G-I-N-D
NC: S-B-C-U-F-H-I-N-D
NFP: S-T-E-G-J-K-L-M-D
Shortest Path: S-T-G-I-N-D
GLAR: S-T-E-I-G-I-N-D
A*: S-T-G-I-N-D

Figure 2.20: Path selections with different greedy algorithms (adapted from **[31]**)

The MFR, CR, and Greedy techniques are further amenable to designs that reduce hop-count by the capability to forward packets over two (or more) hops [30], albeit with increased overhead. Table 2.1 shows comparison of A* technique in VANET environment with the other greedy techniques.

Table 2.1: Comparison of existing greedy forwarding techniques with the A* technique

Technique	**Approach**	**Technique contrast with A* approach in VANET**
Most Forward within Radius (MFR)	Achieves maximal forward progress to the next- hop. Reduces total hop-count.	A* achieves globally optimal shortest path. Minimizes the total distance travelled.
Nearest with Forward Progress (NFP)	NFP aims at transmission energy efficiency by choosing next-hop that is closest, but with forward progress.	Energy rationing not an issue in VANET, hence A* method is agreeable in it.
Greedy basic	Greedy chooses next-hop neighbour closest to the destination at each stage.	A* chooses least distance coverage from currently forwarding node, through the next-hop, to the destination at each stage.
Compass Routing (CR)	CR technique aims at forwarding closely in the direction of the destination, and minimizing the spatial distance a packet travels.	A* chooses path and direction that results in optimally shortest path from source to destination.
Nearest Closer (NC)	NC chooses next-hop that is closest in distance to the forwarding node, with the aim of minimal energy consumption at each hop	Energy rationing is not an issue in VANET; hence A* method is appropriate in it.

In general by using any one of the techniques, the next-hop chosen shall always be with progress further from the forwarding node and closer toward the destination at each hop. This can be represented by the following greedy algorithm.

- choose next-hop as node that is closer to the destination (using some heuristic measure *h*)

A heuristic *h(n)* is defined as admissible if for every node *n*, $h(n) \leq h^*(n)$.

h(n) is the estimated cost to reach the goal

h(n)* is the actual cost to reach the goal

The straight-line distance measure is admissible in the heuristic of each of the above-mentioned techniques, including the A* algorithm.

The A* function *f(n)* = *g(n)* + *h(n)*, an extension of Dijkstra's greedy algorithm, uses the best-first search method to find the least cost path from a start node to a target node on a given terrain map [74]. We discuss the A* forwarding approach next.

2.7 The A* algorithm method

We state the following regarding the A* algorithm in a VANET type of scenario:

If *h(n)* is the estimated linear distance from any node *n* to the destination,

And if *h*(n)* is the actual optimally achievable hopping distance, which may be non-linear from *n* to the destination,

Let *g(n)* be the true distance from the source to node *n*.

The A* search function is defined as *f(n)* = *g(n)* + *h(n)*; *h(n)* = *0* if *n* is the destination.

The A* algorithm is popular for solving optimal path problems over marked terrains. The popularity of A* derives not only due to its efficiency in path finding, but also its adaptability and extensibility to different domains of search problems [33][34]. It has been engaged in robotics, networks, biology, database systems and games [33][34]. An example is the Multi-Agent A* (MAA*) [83] algorithm for solving decentralized partially-observable Markov decision problems. MAA* computes optimal plans for a cooperative group of agents that operate in stochastic environments such as multi-robot coordination, network traffic control, or distributed resource allocation. The algorithm uses heuristic probability factors and decentralized control theory functions in its A* enhanced form. Another example is in ITS systems where A* methodology has been applied to time-dependent shortest path problem in dynamic networks, e.g. for the computation of fastest path to guide truck drivers on routes that minimize travel time [84][35]. Road traffic conditions such as traffic density, speed of travel, geographic obstacles, etc. are factored as heuristics into the A* algorithm to process the shortest path. A* refines path-search in ways that make it to outperform the generalized Dijkstra's algorithm [35]; and its variants implement search strategies that is effective and adaptable [33][32]. Although the Djikstra's greedy algorithm has been commonly applied in routing methods, yet our literature review did not reveal any specific case where A* has been tested in the VANET environment type. Therefore this study would also verify the effectiveness of A* path search over the terrain of non-static grids.

2.8 VANET simulation

The use of simulation tools in data communication networks study is a matured practice [14]; however the development of MANET simulation is yet budding [43]. Especially, VANET simulation methods and tools are still in the formation stages and are yet unstandardized across the research arena; hence researchers have enunciated the importance of faithfully representing the complexities of its environment with valid modelling [85].

The simulation option has been widely adopted in VANET research as against analytical or real-life test-bed experiments. The problem with the analytical approach to VANET study is the intricate level of complex mathematical equations, assumptions, and parameters that will be involved in representing the multifarious interacting phenomenon in such an environment; much of which are even yet to be well discerned [53][43][14]. Real-life experiments on the other hand are resource intensive, very expensive and quite challenging to conduct or reproduce for controlled studies [4][85]. Nevertheless there have been some limited test-bed evaluations within projects that are supported by consortiums of auto industries and Intelligent Transport Systems (ITS) departments, e.g. VII, CVIS and Car2Car [4]. The academia on its part is promoting the development of appropriate VANET simulators, and these are fast evolving.

Fig. 2.21 describes a maturation timeline in VANET simulator development, showing that lately the practice is to couple mobility simulator with network simulator to realise the VANET model. There have existed matured communication network simulators as well as matured mobility generators developed in the Computer Networking and the ITS communities respectively [86]. Thus, some authors have recommended the coupled use of these highly validated network and mobility tools as a standard; however, their interoperability at the unsuited interfaces' level has been another challenge. On another hand, some authors have raised the objection that conventional network simulators are not adequate for utilization in VANET due to the extraneous effects that mobile transmitting-nodes exhibit; e.g. the Doppler effect that is inherent in moving bodies, which should be incorporated at the level of radio channel modelling [39][85]. Hence, VANET simulation practice is yet imperfect, although there is substantial progress made.

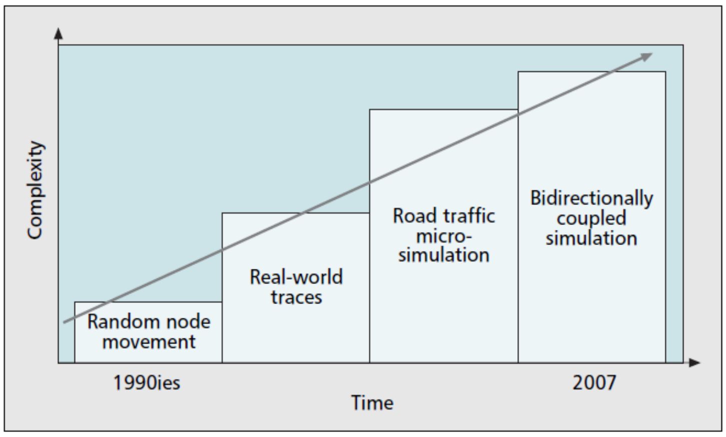

Complexity
Random node movement
Real-world traces
Road traffic micro-simulation
Bidirectionally coupled simulation
1990ies
2007
Time

Figure 2.21: Evolution of mobility modelling strategies and techniques in VANET research [87]

2.8.1 Mobility generation issues in VANET research

Fig. 2.21 shows the evolution of mobility generation and integration in the VANET network simulation environment. In a recent study that polled realistic GPS traces of a few thousand cars over Shanghai city and using the NS-2 network simulator, it was found that the routing protocols Epidemic, AODV, GPSR, and MaxProp performed poorly under differing mobility configurations with realistic modelling features such as vehicular density, traffic load, TTL, transmission range, and propagation models [88][68]. The poor performance was attributed to the uncovering of hidden realities of the VANET dynamics; and this adds to the doubts cast over the efficacy of simulators formerly used and the validity of associated results that have been reported in the VANET literature [43].

The initial and widely adopted mobility generation method (Fig. 2.21) in MANET was the random waypoint model and its variants such as the random direction model; where every node periodically changes location by randomly picking a velocity and a destination throughout the simulation duration [89][85]. Johnson and Maltz [12] had used the Random Waypoint Packet Level Network Simulator for evaluating the DSR (see section 2.1.4.3) routing protocol design. Although this approach is effective for simulating MANET, it obviously does not portray VANET mobility on traffic lanes realistically [90]. The VANET mobility strategy later progressed to the practice of importing real trace files into the network simulation environment; however, the parameters in these snap files are not disposed to manipulation. Subsequently, the practice of coupling network simulator with mobility generator came up; which however is bedevilled with the interface incompatibilities. The contemporary VANET simulator design drive is aiming for a fully integrated system of the mobility generator and network simulator.

2.8.2 Mobility simulators

Fig. 2.22 shows the taxonomy of some simulation tools in use in the VANET community; grouped as mobility generators, network simulators and integrated VANET simulators. Proprietary software, such as TSIS-CORSIM, Daimler-Chrysler Farsi, VISSIM, QualNet and OPNET are not included in this taxonomy as they have been difficult to procure for evaluation [38]. Researchers have used diverse simulators to validate their work, which implies that the relative quality of the published VANET routing protocol designs is not really known [14][57]. On the other hand it has been found that routing protocols' evaluation outcomes depend heavily on the mobility model used [68][37].

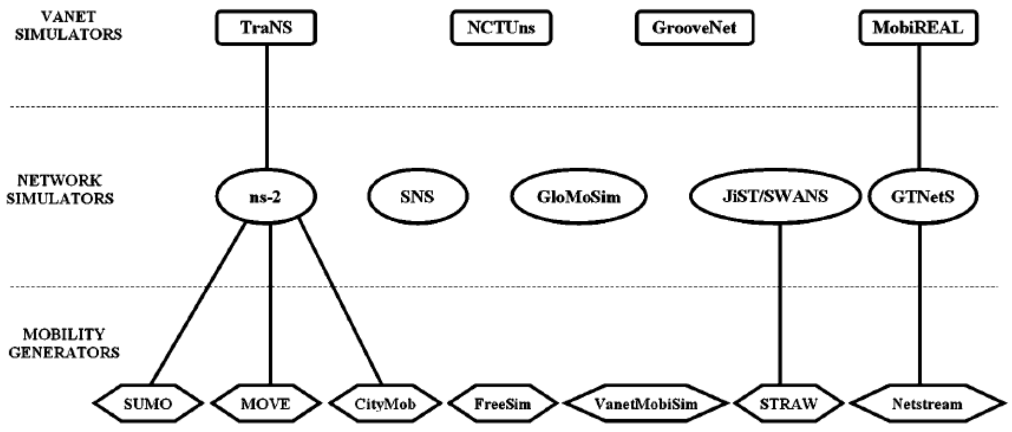


Figure 2.22: VANET simulators [38]

In the transportation science discipline, mobility is largely classified into two basic areas: the macroscopic and the microscopic views [89][91]. Macroscopic model of mobility depicts mass traffic features and effects, such as density, mean velocity, number of lanes, traffic lights, etc. The microscopic mobility model on the other hand depicts discrete entities of vehicles, describing characteristics such as position, acceleration, overtaking, driver behaviour, etc. Both the macroscopic and microscopic models are significantly required in a VANET simulator [38][89]. Generally, mobility modelling classification include the following [92]:

- Stochastic models (e.g. Random waypoint model)
- Traffic stream models
- Car-following models (e.g. Intelligent Driver model) – a car-to-car distance control schema
- Flows-interaction models

We proceed to discuss some of the mobility generators shown in Fig. 2.22.

2.8.2.1 STreet RAndom Waypoint (STRAW)

STreet Random Waypoint (STRAW) [90][85][89] mobility simulator was developed in response to the need for a VANET model that creates road topolography based on real imported TIGER (Topologically Integrated Geographic Encoding and Referencing) street maps in the United States region. STRAW implements the car-following model that depicts features such as street lanes, intersections, stop signs, stoplights, and traffic congestions. Although it implements other traffic control mechanisms such as acceleration and deceleration, yet it lacks the lane-changing feature. STRAW works specifically in conjunction with the Jist/SWANS network simulator.

2.8.2.2 Simulation of Urban Mobility (SUMO)

Simulation of Urban Mobility (SUMO) generator is a powerful micro-mobility traffic simulator with lane-changing features. It is portable on most operating systems, and have been popularly used for VANET research [85][89], particularly in conjunction with the NS-2 traffic simulator. SUMO is written in C++, and contains several extensions contributed by the user community. It is designed to handle large environments of about 10000 streets, and can simulate traffic in different parts of the world using the *openstreetmap* [38]. MObility model generator for VEhicular networks (MOVE) is built on top of SUMO with enhanced user-friendly features and adaptation to make it portable to some other network simulators [85].

2.8.2.3 VanetMobiSim

VanetMobiSim mobility generator is specifically designed for VANET simulation; based on CanuMobiSim which is a flexible framework for MANET modelling [85][38]. It features macro and micro characteristics including the Intelligent Driver Model and Mobility (IDM/MOBIL) standard, with capacity for the management of traffic incidents. VanetMobiSim is written in Java, and can produce mobility traces in various formats for different network simulators, including NS-2, GloMoSim, and QualNet. However it does not regenerate feedback effects, which severely limits its use in realistic VANET simulation [4]. Regrettably also the detail of its implementation is not open, while its development has been discontinued.

2.8.3 Network simulators

The following network simulators are some of the most widely used among the MANET and VANET communities [85].

NS-2

The NS-2, a hitherto widely used network simulator [85][89] has been replaced with NS-3, albeit with no backward compatibility. However, the newly introduced NS-3 lacks ready support for wireless ad-hoc networks, and is difficult to learn and use compared to the GUI-based simulators [93]. NS-2 is written in C++ and Tcl as a modular discrete-event simulator, which has acquired several extensions and revisions. It models several network features and protocols, including 802.11 MAC and PHY elements [38]. A disadvantage of NS-2 is that it has grown highly complex, thereby it hardens the execution of mobility simulators in the framework; while in average conditions, it permits up to only a few hundred nodes running as it consumes a lot of memory and CPU resources [85]. Several mobility simulators couple freely with NS-2 for simulating VANET, including SUMO, MOVE and CityMob.

Jist/SWANS

Java in Simulation Time (JiST) is a general-purpose high performance discrete event engine buoyed by the Scalable Wireless Ad-hoc Network Simulator (SWANS). SWANS is designed basically for MANET [85][38]; but it also couples with the STRAW mobility generator for use in VANET. SWANS is equal in capabilities to NS-2 and GloMoSim, while it exhibits much more efficiency in resource use with the capacity to simulate beyond 10,000 nodes. However, the SWANS project is no longer active.

GloMoSim

Global Mobile Information System Simulator (GloMoSim) is a library-based parallel discrete-event simulator for wireless networks [85][38]. It can work in combination with STRAW as well as VanetMobiSim mobility generators; and has capacity to simulate several thousand nodes. GloMoSim has given way to the commercialized version named Qualnet.

2.8.4 Vanet simulators

Any suitable VANET simulator should incorporate a tightly integrated form of mobility generator and network simulator together with a radio channel modelling [87][90]. But some authors have expressed the view that tight coupling restricts the possibility of modular improvement to the respective models [86].

2.8.4.1 TraNS

Traffic and Network Simulation Environment (TraNS) is a coupling of NS-2 and SUMO, to realise the first VANET simulator with feedback impact on mobility; although not the other way round [85][38][4]. TraNS parses the output of SUMO into NS-2 environment continuously during runtime using TraCI, a general-purpose interface framework that is also developed by the TraNS' authors. Maintenance of TraNS has however been suspended since 2008 [85].

2.8.4.2 Veins

The Vehicles in Network Simulation (Veins) is an open source VANET specific simulator [94]. Veins is a product of the bidirectional coupling of SUMO mobility generator and the OMNeT++ network simulator suite elements, comprising of INET protocols stack with the MiXiM radio propagation framework [85]. The coupling of SUMO and OMNeT++ in Veins is not tight, but it supports feedback reactions both ways. As at present, not much work exists to validate the effectiveness of the Veins simulator.

2.8.4.3 NCTU-ns

National Chiao Tung University Network Simulation (NCTU-ns) was for some time an open source software; but has turned commercial since 2011 under the new name of EstiNet [85]. NCTU-ns models the complete IEEE WAVE (Wireless Access in Vehicular Environments) architecture as well as several wireline and wireless standards including IEEE 802.11b, 802.11e, 802.11p, 802.16d, etc. It is written in C++; and features quite an advanced GUI. NCTU-ns incorporates mobility models that include random walk, car-following-IDM with lane-changing and intersection management. The network and mobility simulation modules in NCTU-ns are bidirectional and tightly integrated. The EstiNet is a multifunctional network simulator and emulator having link to the real Linux TCP/IP protocol stack in its engine, which operates on a discrete-event mechanism with *kernel-re-entering* capacity. It runs on Fedora OS platforms; with capability of instantiating up to 4096 nodes within a simulation run, although this size may be increased upon effecting some specified modifications. The EstiNet simulator development is an ongoing project.

2.8.5 The choice of simulator for VANET study

So far there is no de facto simulator adopted for VANET research, but some directions for the development of such has largely emerged [85], which is due to improved understanding of its underlying requirements. There are some contemporary VANET simulators that have appeared popular or widely patronized, including NS-2 and NCTU-ns/EstiNet [38]. NS-2 requires coupling with mobility generators to yield secondary simulators such as TraNS; while NCTU-ns, as shown in Fig. 2.23, is depicted as strong on both sides of mobility and network simulation [4]. Furthermore, NCTU-ns exhibits a lot of advantages over traditional simulators such as NS-2 or OPNET [38], which is due to its novel kernel re-entering technology. Both NS-2 and GloMoSim are poor in modelling very large networks while JiST/SWANS is harder to use than others [38]

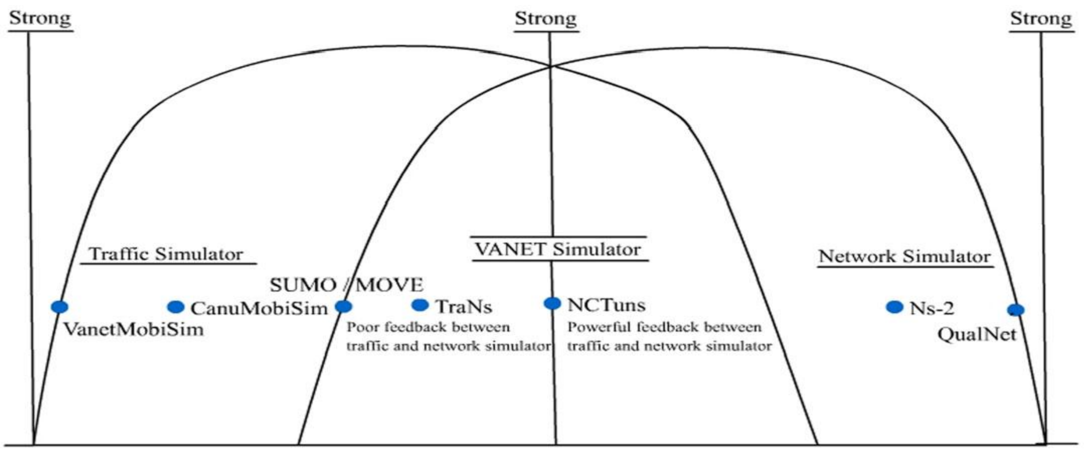


Figure 2.23: Strength of Traffic, VANET, and Network simulators [4]

We summarise features of three VANET simulation environments for evaluation in Table 2.2.

Table 2.2: Comparison of TraNS, Veins, and EstiNet simulators [38][85][89]

Characteristic	**V a n e t S i m u l a t o r**		
	TraNS	**Veins**	**NCTU-ns / EstiNet**
Network Simulator Capability	NS-2 Ease of use – Moderate Ease of setup – Moderate	OmNet++ Ease of use – Hard	NCTU- ns / ESTINET Ease of use – Hard Ease of setup – Hard
Mobility Generator Capability	SUMO Random and manual set up routes model Car following Ease of setup – Moderate Ease of use – Hard	SUMO	NCTU- ns / ESTINET Random and manual set up routes model Microscopic, space-continuous and time-discrete Lane models Multi-lane streets with lane-changing Car following
Coupling of Mobility And Network Simulator	Unidirectional coupling. Information exchanged in communication protocols can influence the vehicle behavior in the mobility model.	Bidirectional coupling	Tightly integrated coupling The design of the general purpose simulator includes several libraries such as the one for VANET
Applications Development Suitability	Routing Protocol - suitable MAC Protocol – suitable Safety Applications - unsuitable Info dissemination - avoid Traffic Management -avoid Internet Access – suitable	Routing Protocol - suitable MAC Protocol – recommended Safety Applications - unsuitable Info Dissemination - avoid Traffic Management - avoid Internet Access - suitable	Routing Protocol - unsuitable MAC Protocol – suitable Safety Applications - recommended Info dissemination - unsuitable Traffic Management - unsuitable Internet Access - recommended
Ease of Set Up	Moderate	Hard	Moderate
Ease of Use	A lot of manual parameter inputs are needed for TraNS		Mostly GUI guided parameter selection dialog
Availability	Development discontinued since 2008	Commenced mid-2000s, and being developed by community. Not much published work to validate its use.	Commenced early 2000, and being actively improved. Much used in published works.

Generally, the criteria for selecting a VANET simulator should include important questions of [4][85]:

- usability features
- complete characterization of VANET, in particular mobility modelling, and
- the type of application that is being investigated

NCTU-ns (aka EstiNet) is deemed suitable on the conditions of usability and fuller characterization of VANET, although it attains a limited recommendation for simulating routing protocol study [85]. Nevertheless, we choose to use EstiNet because of its availability, while we work within the frames of its limitations in this study. The EstiNet simulator/emulator suite is a multifunctional product containing specific network libraries including VANET's. Table 2.3 describes features of the EstiNet package.

Table 2.3: Features of the EstiNet simulator/emulator (adapted from [95][96])

EstiNet Simulator Engine 8.0 – Basic framework	
Fixed network library	Ethernet hub / switch / router, Optical networks, RIP I/II, OSPF
Wireless network library	IEEE 802.11 (a)(b)(g) ad-hoc and infrastructure mode networks, mobile ad hoc networks, IEEE 802.11(b) wireless mesh networks, Mobile IP, GPRS networks, channel propagation models
Media streaming library	Media streaming
QoS Library	RTP / RCTP / SDP /QoS Diffserv , IEEE 802.11(e) QoS WLAN
EstiNet can incorporate real-life network applications during a simulation/emulation.	
EstiNet can incorporate real-life Linux TCP/IP protocol stacks during a simulation/emulation	
OptionalAdd-onModules	
Emulator	IP real network emulation Library
Fixed network advanced features	MPLS / VPLS / VLAN / VRRP / RSTP / STP / MSTP / Static Route / RIP/ OSPF / BGP / IS-IS
Wireless network advanced features	IEEE 802.11n (EDCA , A-MSDU , A-MPDU , Block Ack and Primary & Secondary)
VANET	Wireless vehicular networks Library* (*with source code)
Open Flow	Open Flow switch 1.1.0 / 1.0.0
LTE	LTE R.9 , 3GPP TS 23.401 / TS 24.301 /TS 36.331 / TS 36.322 / TS 36.231 / TS 36.211 / TS 36.212 / TS 36.213
OS Platform	
Fedora Linux (open source)	

The emulator module of the EstiNet package allows the integrative use of the simulator with a linking to real life applications, operating systems and devices. EstiNet is reputed to have been in patronage by over 20000 customers or institutions in about 144 countries, and has over 900 published papers that support its credible use; including several of them that are involved with VANET-related study. EstiNet Technologies Inc. runs an EstiNet University Programme (EUP) license for qualified institutions upon relevant request to the organisation. The architectural features of the EstiNet simulator/emulator in Fig. 2.24 show its components and general features.

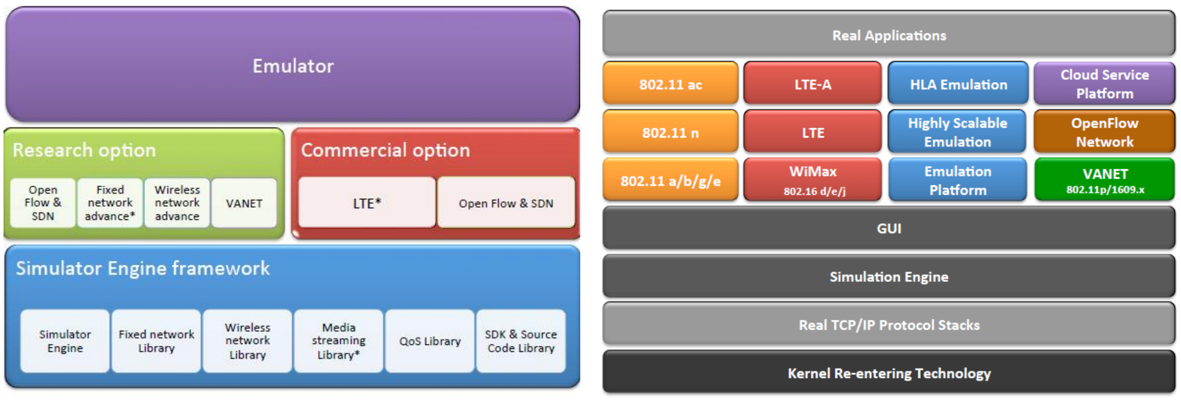


Figure 2.24: Features of EstiNet architecture [97]

2.8.6 Simulation environment

The following conditions should necessarily hold in the simulation environment of vehicular nodes with data packets.

- Nodes are in constant mobility state
- Each node knows its current position coordinates
- Each node is aware of its immediate neighbors' positions
- The destination's current position is known
- Vehicular traffic dynamics
- Generation of communication events

Table 2.4 compares some general simulation parameters used in the stated protocols' development evaluation, including GPSR, AODV, etc. The minimal packet size of 64 bytes that was used in GPSR column was rather chosen, according to its author, because larger packets tend to monopolize the communication channel and cause queue overflow at the MAC layer [98], or even induce excessive collisions [11]. We referenced these parameters in our choices that are shown in Table 4.1.

Table 2.4: Simulation parameters comparison

Simulation Environment	GPSR [21][98]	AODV [11]	GLAR [19]	Epidemic, AODV, GPSR, MaxProp [88]
Network simulator	NS-2			NS 2.34
Simulation time	900s	600s	600s	14,760s
Simulation area	1500m x 300m, 2250m x 450m, 3000m x 600m	50m x 50m, 100m x 100m, 150m x 150m	600m x 600m	131km x 89 km
MAC protocol	IEEE 802.11			IEEE 802.11 DCF
Traffic application	CBR			CBR
Number of nodes simulated	50, 112, 200	50, 100, 500, 1000	30	500 - 1000
Traffic load (packet size)	64 bytes			20-100 KB
Transmission range	250m	10m	100 m	100 – 500 m
Propagation model	Random waypoint mobility model	Random	Random waypoint mobility model	Two ray ground shadowing
Speed	0 - 18 m/s (64.8km/hr)	0.4-0.8 m/s	20 – 80 km/hr	

3.0 METHODOLOGY

In this section we describe the various functional parts of the S* routing method as well as the relevant algorithms for its implementation. We first explain the classic A* model and its application to the VANET context, and then we outline the S* module's components. In the second part, we show the routing protocol's basic interaction structures for forwarding packets, and we show the core parts of the greedy selection code.

3.1 The A* algorithm function

In the classic A* algorithm applications, the objective usually is to achieve optimal path search over a graph, going from a start node to a goal node. A* was originally proposed by Hart, Nilsson, and Raphael [32] as a best-first search algorithm that always expands the most promising node n based on the evaluation function $f(n) = g(n) + h(n)$. What sets A* apart from a greedy best-first search is that it also takes the value of the path already covered into account, the $g(n)$ part, which is the cost from the starting point and not simply the local cost from the previously expanded node [99]. The function $g(n)$ is the cost of the path from the start node s to a current node n along the paths found so far in a search tree. The function $h(n)$ is an heuristic estimate of the cost of the cheapest path from n to the goal node. During a path search, A* uses the function f to evaluate its candidate next-hop nodes and selects the best, i.e. the node that yields the lowest f value.

The A* algorithm is presented as follows (Fig. 3.1):

```
Algorithm A*

  (1)     Put the start node $s$ into OPEN.
  (2)     IF OPEN is empty THEN exit with failure.
  (3)     Remove from OPEN and place in CLOSED a node $n$
          for which $f$ is minimum.
          (Resolve ties for minimal $f$ value,
          but always in favour of any goal node).
  (4)     IF $n$ is a goal node THEN exit successfully,
          with the solution obtained by
          tracing back the pointer from $n$ to $s$.
  (5)     ELSE expand $n$, generating all its successors,
          and attach to them pointers back to $n$.
     FOR every successor $n'$ of $n$ DO
     (5.1) IF $n'$ is not already in OPEN or CLOSED
        THEN estimate $h(n')$,
          and calculate $f(n') = g(n') + h(n')$,
          where $g(n') = g(n) + c(n, n')$ and $g(s) = 0$,
          and put $n'$ into OPEN.
     (5.2) IF $n'$ is already in OPEN or CLOSED
        THEN direct its pointer along the path
          yielding the lowest $g(n')$.
     (5.3) IF $n'$ required pointer adjustment and was in CLOSED
        THEN reopen it.
        END FOR
  (6)     GO TO step 2.
```

Figure 3.1: The A* path-search pseudocode [32][34]

We restate the following functions:

- Best First search: $f(n)=h(n)$, (3.1)
- Uniform Cost search: $f(n)=g(n)$, (3.2)

- A* search: $f(n) = g(n) + h(n)$, (3.3)

Note that $f(n) = h(n)$ is the Greedy selection function that is used commonly in the various geographic forwarding routing protocol designs that we reviewed in sections 2.3 and 2.4.

The A* function $f(n) = g(n) + h(n)$ is the estimated cost of the cheapest solution going through n, thus in VANET distance measures,

- $g(n)$ = actual distance from the source node s to the currently forwarding node n.
- $h(n)$ = estimated distance from n to the destination.

In the VANET implementation of A* the accumulative OPEN and CLOSED global queues shall not apply because each node implements the algorithm locally, before forwarding packet(s) unit together with the relevant parameters to the next-hop. Moreover, any OPEN and CLOSED memory implementation in VANET unavoidably quickly becomes outdated in the face of a rapidly changing topology.

3.2 The S* routing module design

The classic A* computes and returns the result of the complete path links from a source node to a goal node over a static graph. However, the VANET graph is not static, requiring that the implementation of the algorithm in reinitiates at every forwarding node n, and returns the path to the next-hop. Hence, the complete path that a packet traverses is the connection of the *forwarding points,* i.e. the locations of the vehicular nodes as at the times of forwarding. We give the term *Successive A** to this forwarding algorithm because every forwarding node initialises an instance of it as a packet is forwarded from node to node, and we use the symbol *S** for the convenience of referring to it.

The definitive objective of the S* routing method is to realise optimal geographic forwarding of packets in VANET using the A* algorithm approach. The S* routing module uses some terminologies and concepts from the GLAR model. We first describe the GLAR model, and then we make an application of A* to formulate S*.

3.2.1 The GLAR route discovery model

The Greedy Location-Aided Routing (GLAR) [19] protocol design has been described in section 2.4.6. Every node in GLAR maintains a routing table similar to that of classic protocols such as AODV [11], and it is only when the route to a destination is not known by any neighbour node that the RREQ broadcast is initiated by the source node to derive the required path. The peculiar feature of the GLAR protocol model is its use of a source to destination computed *baseline*, as shown in Fig. 3.2, to direct route discovery effort along this reference line by choosing next-hops that are closest in distance to it.

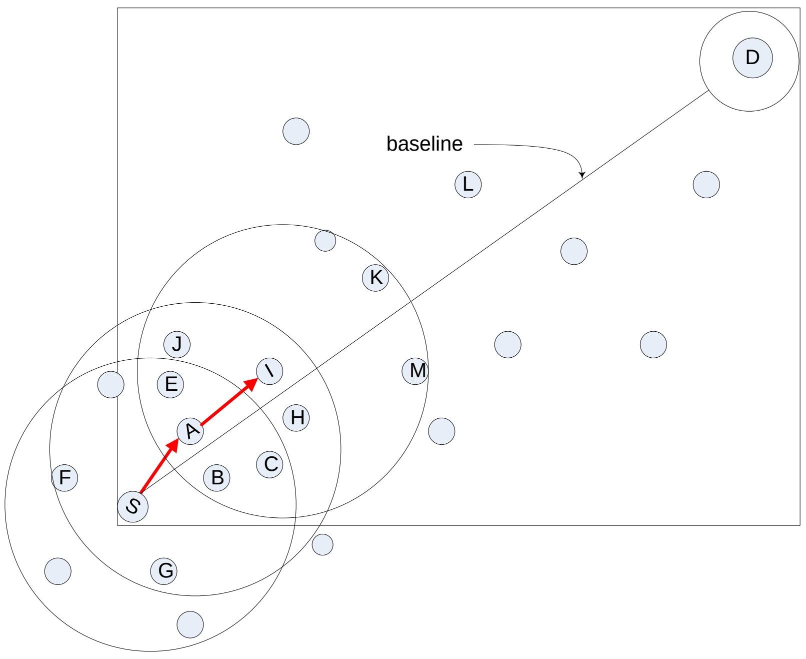


Figure 3.2: GLAR protocol model**[19]**

In Fig. 3.2, the circle around node *D* is the *expected zone* of reaching it, which is defined by the product of the node's velocity and time interval - between t_0 when the node's location is first pinpointed and time t_1 when the route discovery action is initiated. The rectangular area is the *request zone* that delimits (and reduces) the area of path search beyond which outlying nodes discard RREQ broadcast messages. From the source node *S*, the GLAR algorithm selects node *A* as the next-hop; and then node *A* selects node *I* and so on; which ensures that every successive next-hop has a higher progress value of distance *DIST* (Fig 3.3.) away from the source node *S*. The parameter *VDIST*, i.e. vertical distance, as shown in Fig. 3.3, functions to determine the proximity of the nodes to the baseline.

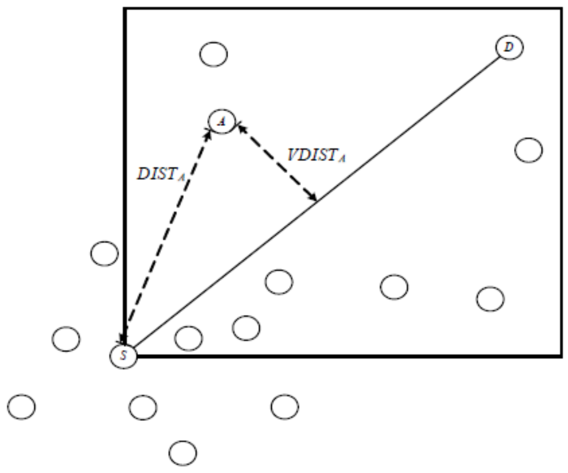


Figure 3.3: An example of *DIST* and *VDIST* **[19]**

The equation of the line *SD*, i.e. the baseline, is determinable by

$$(x_d - x_s)(y - y_s) - (y_d - y_s)(x - x_s) = 0 \qquad (3.4)$$

And when the values of coordinates of *S* and *D* are assigned into equation (3.4), it results in the equation of a line, $ax + by + c = 0$ (1.3)

Therefore within the search region, for any node A with coordinates (x_A, y_A) , its distance to the baseline is:

$$VDIST_A = \frac{ax_A + by_A + c}{a^2 + b^2} \qquad (3.5)$$

3.2.2 The S* forwarding technique for VANET routing

We assert that in the VANET environment, a single persistent baseline as used in GLAR [19] cannot be relied upon to accurately route packets, as vehicular nodes' mobility rate is much higher than what is obtainable with common MANET devices. A persistent baseline view is much similar to charting an end-to-end session in conventional networks, a practice that has been shown to be infeasible in VANET. We employ baselines generation at every forwarding point to aid local decision-making and more precise routing. Moreover, the S* approach will involve packet swinging (described in section 2.3.7) where each packet independently progresses towards moving next-hop targets and destination through regular re-computations of its path.

The S* forwarding technique is then implementable as follows:

- Every node possesses a copy of the enhanced A* program running on it in the S* application ensemble.
- Each node *n* periodically beacons every neighbour node n' within its transmission range and acquires their respective identity and geographical position information – for its maintenance of neighbour list.
- Any packet(s) forwarding node *n* evaluates its neighbour list, from which it makes choice of a next-hop node n'.

- The next-hop choice shall be a node within the positive advance region relative to the forwarding and destination nodes' positions, as depicted in Fig. 3.4. A node *n'* is in the positive advance region if for a computed Euclidean distance *h* to the destination, then *h(n') < h(n)*.

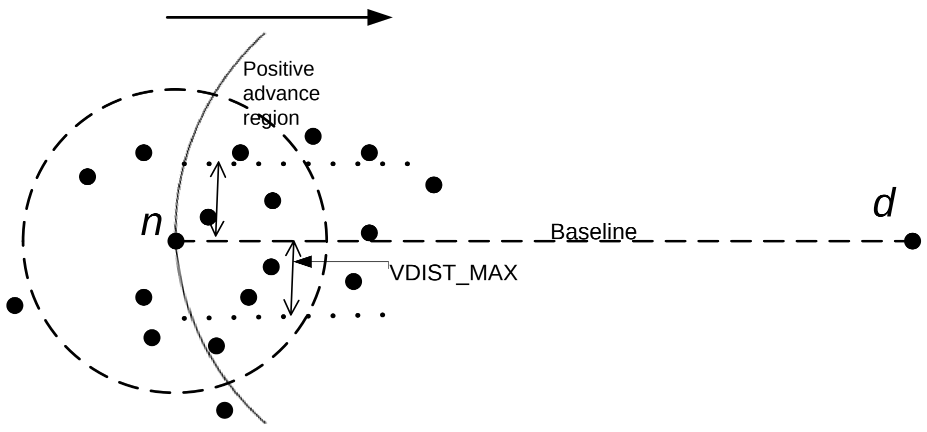


Figure 3.4: Next-hop choice region delineation for a forwarding node *n*

- The eventual next-hop choice is the node *n'* with the least *f* value, where *f(n') = g(n') + h(n')*. In Fig. 3.5, *f(b)* has the least value in the evaluation of nodes *a* and *b* for next-hop choice; hence node *b* is chosen.

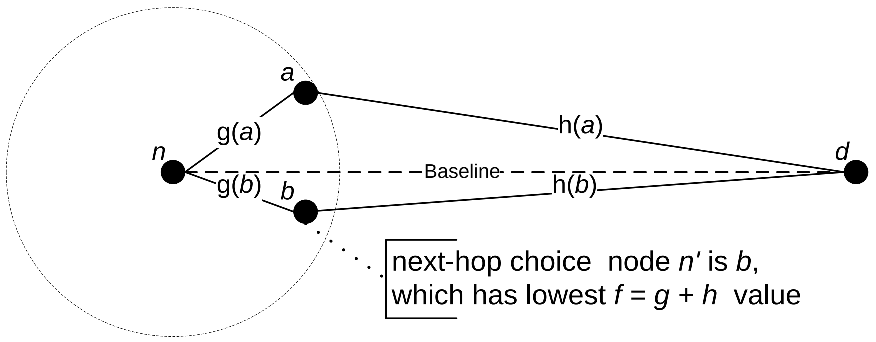

a
g(a)
n
g(b)
b
Baseline
h(a)
h(b)
d
next-hop choice node n' is b, which has lowest f = g + h value

Figure 3.5: Next-hop selection function based on A* algorithm

A distinguishing feature of S* technique is that it operates over extremely mobile grids as compared to the traditional A* that searches for path over static grids. As such, S* does not keep children or ancestors of node items in queues but discards such data as soon as the next-hop is decided. A problem associated with the classic A* is its memory space requirement when very large numbers of nodes' data are to be kept in the global OPEN and CLOSED lists. The memory space problem is eliminated in S* as the memory for handling the neighbour list is minimal as well as local to every VANET node *n*.

With reference to Fig. 3.6; for a source node *s*, a forwarding node *n*, and a next-hop node *n′*, when $f(n) = g(n) + h(n)$, we have $f(n') = g(n') + h(n')$.

Thus $g(n) = c(n, s)$, and $g(n') = c(n, s) + c(n', n)$ for all *n′*

But the cost *g(n′)* for any node *n′* includes the cost so far to *g(n)* for all *n′*.

So $g(n') = c(n', n)$, since *c(n, s)* is a constant value in regard to all *n′*. (Note that *c(n, s)* span, shown in the figure, may be over several nodes that have been previously traversed).

This means we may use just the local cost from *n* to each *n′* as the relevant *g(n′)* cost part.

Hence it is not necessary to keep knowledge of prior paths that packet(s) have taken in the S* implementation, and we do not need to keep any global or persistent OPEN and CLOSED lists.

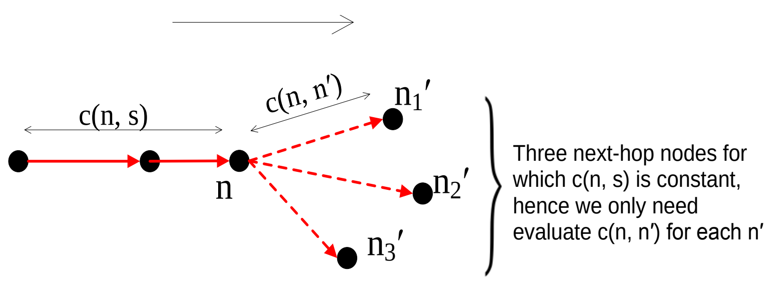

c(n, s)
c(n, n′)
n
n_1'
n_2'
n_3'
Three next-hop nodes for which c(n, s) is constant, hence we only need evaluate c(n, n′) for each n′

Figure 3.6: The g(n) cost model in the S* function

We show the S* next-hop selection pseudocode in Fig. 3.7. The Greedy pseudocode in the GPSR [98][21] forwarding construct is comparable to this, where the main difference is in line 7 that has the Distance(n, n′) as an added function.

```
Algorithm S* GetNextHop (dst_n)
(1)  bestnexthop = n  // self is n
(2)  least_f = 10000000  // a very large value for distance f = g
     + h
(3)  for each n′ in L do
(4)    if n′ == dst_n
(5)    bestnexthop = n′
(6)    break
(7)    distance_f = Distance (n, n′) + Distance (n′, dst_n)
(8)    if distance_f < least_f
(9)    bestnexthop = n′
(10) return bestnexthop
```

Figure 3.7: S* Next-hop Selection Pseudocode

The search-space orthogonal limit selection function

The region within which to select a next-hop (Fig. 3.4) may be restricted to an orthogonal distance *VDIST_MAX* away from the baseline. Therefore:

- For a forwarding node with coordinates (x_1, y_1) and a destination with coordinates (x_2, y_2), the connecting baseline is derived by

  $(x_2 - x_1)(y - y_1) - (y_2 - y_1)(x - x_1) = 0,$ (1.2) and when the values of (x_1, y_1) and (x_2, y_2) are assigned, we derive the straight-line formula $ax + by + c = 0$. (1.3)

- For any qualifying next-hop node n' with coordinates ($x_{n'}$, $y_{n'}$); then for any chosen limit value of *VDIST_MAX*,

$$VDIST_{n'} = \frac{ax_{n'} + by_{n'} + c}{\overline{a^2 + b^2}} \leq VDIST\_MAX \qquad (3.7)$$

An S* forwarding model

The S* model in Fig. 3.8 below extends the GLAR [19] model of Fig. 3.2; which we describe as follows:

- The source node *S* is to send data packets to the destination node *D;* but node *D* is moving in the direction southwards from the top-right corner as indicated in the model.

- At node *S*, the S* routing algorithm makes a next-hop choice among neighbour nodes within its range. Node *A* is selected as next-hop since it would be evaluated to have the least *f* = *g* + *h* value.

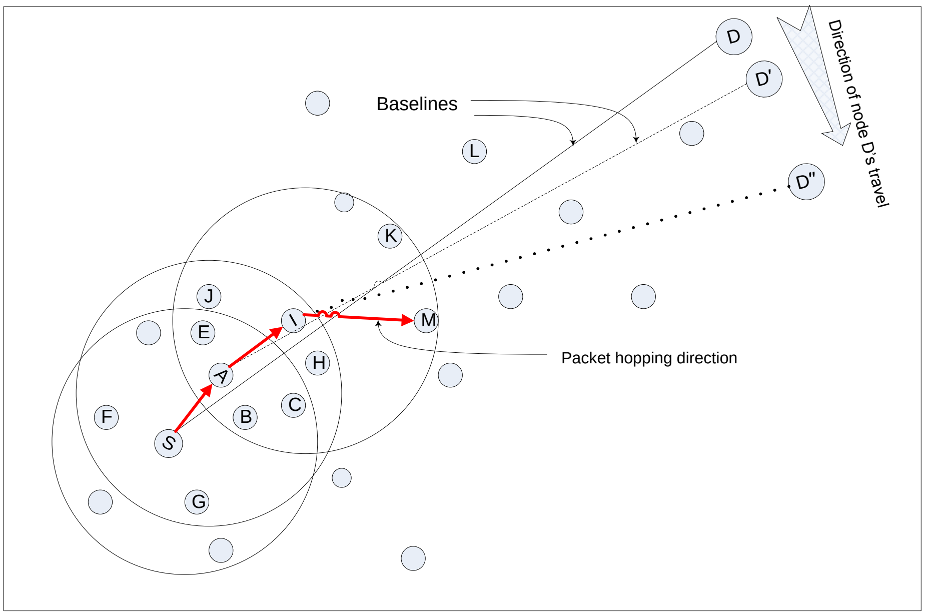


Figure 3.8: S* forwarding model (adapted from **[19]**)

- Subsequently, the forwarding node *A* selects node *I* as next-hop, while node *I* selects node *M*, and so on. The directed edges *S*-*A*, *A*-*I*, and *I*-*M* indicate the packet forwarding path.

  Note that S* selected node *M* rather than *K* that the GLAR scheme (Fig. 3.2) would have selected.

3.3 Simulation algorithms design

Regarding the greedy forwarding simulation for VANET that we implement, we first show simple packets forwarding use-case and collaboration diagrams and then we present a relevant code segment.

In the use-case model (Fig. 3.9), every node that is involved with packet(s) handling is aware of itself as source, intermediate or the destination node. The model shows the three main packets handling functions: *MaintainNeighbourList, DetermineNextHop* and *ForwardPacket*. Every node in the system maintains knowledge of its next-hop neighbours' positions. A node that has any packet to handle receives with it the destination information as supplied by *ForwardPacket*. Then it selects its next-hop by going through neighbour list obtained from *MaintainNeighbourList*. Finally, *ForwardPacket* dispatches the packets. The *SelectNextHop* submodule embeds the main greedy selection code, which may be Greedy, S* or any other technique.

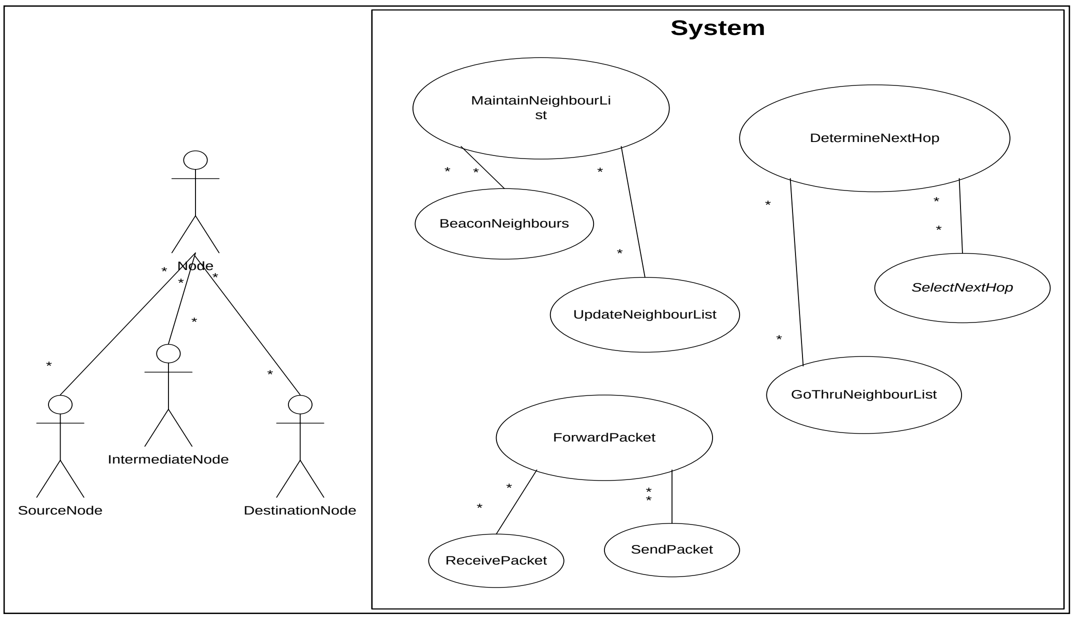


Figure 3.9: The greedy packets forwarding use-case diagram

The cycle of collaboration among nodes is depicted in Fig. 3.10. Normally every node in our simulated VANET context engages in periodic beaconing of neighbours to sustain its neighbour positions' knowledge. As soon as a potential *ForwardingNode* receives packets, it uses the destination information to select a suitable node from *Nexthop Neighbours*, a list of which it maintains. Thereafter it sends the packet(s) to the *NextHopNode*.

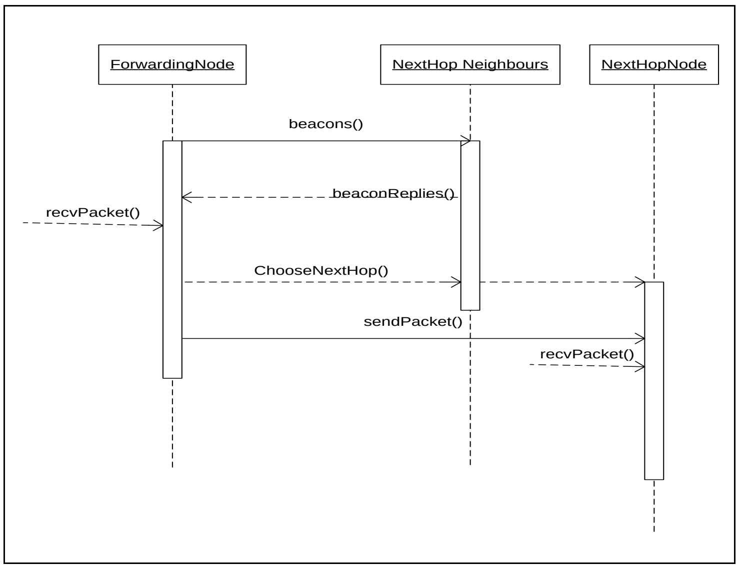


Figure 3.10: The greedy packets forwarding collaboration diagram

We developed the codes for the implementation of both the Greedy and the S* forwarding modules and embedded these in the generic *UserModule* template that is prescribed in the EstiNet simulator. We modified and adapted an AODV routing protocol code copy that runs in EstiNet to implement the greedy forwarding functions. We disabled the routing table entry function for the forwarding of non-AODV packets and replaced that part with our forwarding code. The S* code and the Greedy code that we developed are quite similar; with a difference only on a single line that implements the g part of $f = g + h$ evaluation.

We show the S* forwarding code segment in Fig. 3.9 below, which contains the following sections:

- positive advance element, in line 1765
- neighbour list traversal function, beginning at line 1746

- Greedy or S* selection function (in line 1768, assigns nei_g = 0 for the Greedy version of the algorithm)
- vertical distance evaluation function, line 1760

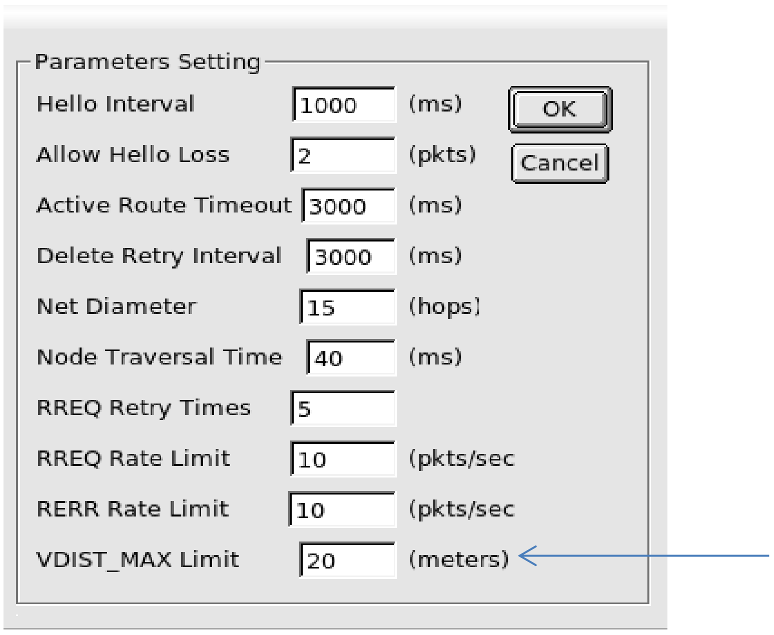
Appendix A displays some more portions of the adapted AODV codes.

Apart from the object file names, the only difference between the S* and the Greedy algorithm programs is in line 1768.

```
1696 ////////////////////////////////////////atanda, ASTAR ENGINE
CODE////////////////////////////////////////
1697 int UserModule02::getnexthopAStar(u_long dst_ip)
1698 {
1699    //initialize to myself
1700    u_long bestnexthop = *mip;
1701
1702    int dst_NID;
1703    double dst_X, dst_Y, dst_Z;
1704

1706    dst_NID = ipv4addr_to_nodeid(dst_ip);
1707
1708    GetNodeLoc(my_NID, my_X, my_Y, my_Z);
1709    GetNodeLoc(dst_NID, dst_X, dst_Y, dst_Z);
1710
1711    //BEGIN Calculate baseline values
1712    double a, b, c;
1713
1714    /* Compute the aX + bY + C = 0 line equation formed by (x0, y0) and (x1, y1). //atanda,
template obtained from obstacles.cc*/
1715         if (dst_Y == my_Y)
1716    {
1717                 a = 0;
1718                 b = 1;
1719                 c = -1 * my_Y;
1720                 if (dst_X == my_X)
1721                         a = b = c = 0;
1722         }
1723         else if (dst_X == my_X)
1724    {
1725                 a = 1;
1726                 b = 0;
1727                 c = -1 * my_X;
1728         } else
1729    {
1730                 b = -1;
1731                 a = (dst_Y - my_Y) / (dst_X - my_X);
1732                 c = my_Y - my_X * a;
1733         } ///END Calculate baseline values
1734
1735    //my estimate straight line distance to destination
1736    double my_h = sqrt(((my_X - dst_X)*(my_X - dst_X)) + ((my_Y - dst_Y)*(my_Y - dst_Y)));
1737    int nei_NID;
1738    double nei_f, nei_g, nei_h;   //For f = g + h
1739    double least_f = 10000000;    //initialize with an infinite big value
1740    double nei_vdist;
1741    double nei_X, nei_Y, nei_Z;
1742
1743    Nei_entry *p_nei = nei_list.getHead();
1744
1745    //BEGIN Traverse list of my neighbors and make selection
1746    while (p_nei)
1747    {
1748       nei_NID = ipv4addr_to_nodeid(p_nei->nei_addr);
1749
1750       if (nei_NID == dst_NID)
1751       {
1752          bestnexthop = p_nei->nei_addr;
1753          break;
1754       }
1755       else //assess for nexthop
1756       {
1757       GetNodeLoc(nei_NID, nei_X, nei_Y, nei_Z);
1758
1759       ///perpendicular distance to baseline, VDST = |((aX+bY+c)/sqrt((a*a)+(b*b)))|
1760       nei_vdist = fabs(((a * nei_X) + (b * nei_Y) + c) / sqrt((a * a) + (b * b)));
1761
1762       nei_h = sqrt(((nei_X - dst_X)*(nei_X - dst_X)) + ((nei_Y - dst_Y)*(nei_Y - dst_Y)));
1763
1764       //positive advance and restrict range of nodes allowed from baseline
1765          if ((nei_h < my_h) && (nei_vdist < VDIST_MAX))  // int VDIST_MAX .h file
1766          {
1767             //Calculate f = distance g + distance h. But for Greedy assign nei_g = 0.
1768             nei_g = sqrt(((my_X - nei_X)*(my_X - nei_X)) + ((my_Y - nei_Y)*(my_Y - nei_Y)));
1769             nei_f = nei_g + nei_h;
1770
1771             if (nei_f < least_f)
1772             {
1773                least_f = nei_f;
1774                bestnexthop = p_nei->nei_addr;
1775             }
1776          }
1777       }
1778       p_nei = p_nei->next;
1779    }
1780    return bestnexthop;
1781 }//getnexthopAStar
```

Figure 3.11: S* forwarding code segment

We designed the dialog box mechanism (Fig. 3.12) for the entry of *VDIST_MAX* (maximum vertical distance to the baseline) values as depicted in line 1765 of the S* code segment (Fig. 3.11); while appendix A3 shows the code that generates the dialog box.



Parameters Setting

Hello Interval	1000	(ms)
Allow Hello Loss	2	(pkts)
Active Route Timeout	3000	(ms)
Delete Retry Interval	3000	(ms)
Net Diameter	15	(hops)
Node Traversal Time	40	(ms)
RREQ Retry Times	5	
RREQ Rate Limit	10	(pkts/sec
RERR Rate Limit	10	(pkts/sec
VDIST_MAX Limit	20	(meters)

Figure 3.12: Dialog-box for setting protocol parameters, showing the VDIST_MAX slot

3.4 Simulation parameters

We carried out the VANET packets forwarding experiments using the EstiNet simulator with the general parameter settings shown in Table 3.1; and which are comparable to those shown in Table 2.4, particularly with that of GPSR. We imported Openstreetmap files, as shown in Appendix B, to simulate road networks depicting four meters wide single-lane tracks in both directions. Conventionally, most studies simulate on two-dimensional space, which we also adopted; however, it is simple to extend greedy forwarding to three-dimensional space [98].

Table 3.1: Simulation Parameters

Parameter	Configuration
Simulation time	600s, 1000s
Simulation area	1700m x 600m; 12000m x 4500m approx.
MAC protocol	IEEE 802.11p
Traffic application	CBR
Number of nodes	10, 15, 100, 250
Traffic load (packet size)	64 bytes
Transmission range	885m
Propagation model / Mobility model	Two ray ground shadowing; random, microscopic, space-continuous and time-discrete, multi-lane, lane-changing and car following
Nodes' speed	0 -18m/s (0 - 64.8km/hour)

Due to hardware limitations, we simulated only up to 250 nodes conveniently. In the AODV's evaluation [11], the author expressed the hardships of running 1000 nodes. In GPSR's evaluation [98], the author simulated 50, 112 and 200 nodes of mobile devices having uniform density of 1/9000m^2, which however would be too dense to describe a VANET situation. The DSR's simulation density was lower at 1/35912m^2 [98]. We simulated node speeds in the range of 0-18 m/s (0-65 km/hour), which is analogous to what obtains in urban areas. We show in Table 3.2 the nodal density ratios of the simulation environments, with example visual snapshots in Appendix C.

Table 3.2: Simulation node densities

Nodes	Region (approx.)	Density
10	1700m x 600m	1/ 102000 m^2
15	1700m x 600m	1/ 68000 m^2
100	12000m x 4500m	1/540000m^2
250	12000m x 4500m	1/216000m^2

The simulator comprises the model of Wireless Access in Vehicular Environments (WAVE) Service Advertisements (WSA) beacon services, by which every node establishes communication channel connectivity with its neighbours. We set the WSA at the minimum level of one per every 5 seconds, to minimize network congestion, as the routing protocol also beacons neighbour discovery packets periodically. We simulated the multi-hop transmission of packets over the network from a single source to a single destination since our primary aim is to determine end-to-end packets delivery success rate; but the evaluation of GPSR protocol had involved 22 sending nodes. The packets' traffic that we used is of the udp type, which is depicted as stg/rtg in EstiNet with parameters as shown in Table 3.3. We adjusted the FIFO packets queue from the default 50 to 1000 to minimize packets dropping occurrences. Appendix D displays the protocols stack in EstiNet among which we plugged the 'user_module' of Greedy and S* forwarding modules.

Table 3.3: Simulation packet specifications

Parameter	Source node settings	Destination node settings
User datagram protocol (udp)	stg -u	rtg -u
Packets size	64	
Simulation time	1000	
Port	-p 8001	-p 8001
Target address	xx.xx.xx.xx	

We used transmission range of 885m; having transmission power of 28.80814dbm, which is the default in EstiNet. GPSR [98] evaluation simulated the range of 250m, while for GPSR-BB [100] it was 250m and 1000m ranges. The US Dedicated Short Range Communications (DSRC) range specification is up to the range of 1000m; nevertheless using large transmission ranges increases the probability of packet collisions in a busy network [80]. The *VDIST_MAX* (i.e. the maximum orthogonal distance to the baseline settings) parameter is set at 1000m default.

3.5 Simulator limitations in the study

The EstiNet simulator runs on remote servers, while the simulations files are quite large; hence, it requires high-bandwidth unbroken Internet connectivity as well as high-performance hardware to process jobs. Thus, the inadequate availability of these resources meant that sometimes a job runs for as long as 24hrs. We also could not process larger than 250-node job files, neither could we perform multiple re-runs for presentation due to limited study period.

3.6 Analysis data characteristics

The EstiNet simulator provides three basic forms of output data: (1) graphs, (2) GUI animation, and (3) packets' trace. We majorly used the graph outputs to analyse and compare packets forwarding and delivery performances. The GUI animation displays the motion of nodes and packets. Packets trace shows the execution characteristics of each packet piece, which we needed not to present in this report.

The confidence interval value for the simulation runs could not be mathematically determined; but what we did instead is to perform repeated runs of some simulation files and compare the different outputs for consistency. We show in Appendix E an example set of outputs obtained from the running of a file.

The simulation results' graphs display the number of unicast incoming or outgoing packets delivery success rates. The incoming packets data is read at the destination node, while that of the outgoing is at the sending node. The graphs show packets delivery levels over the simulated period of 1000 seconds.

4.0 SIMULATION RESULTS & EVALUATION

In order to evaluate the performance of the S* packets forwarding technique in the VANET environment and in comparison to the basic Greedy technique, we simulated the algorithms on the EstiNet simulator. In addition, we simulated and compared the location-aware forwarding method with the table-driven forwarding of Ad-hoc On Demand Vector (AODV) protocol.

4.1 Performance of the S* forwarding technique

The simulation results of the basic Greedy and the S* forwarding techniques are presented as graphs and compared in this section. We show results for the different numbers of nodes simulated, including 15, 100, and 250.

4.1.1 Comparison of Greedy and S* unicast incoming packets

Fig. 4.1 (a) - (i) reflect the number of unicast incoming packets received at the destination node over the network sizes shown and for the Greedy and the S* forwarding methods.

Figure 4.1 (a) - (i): Comparison of Greedy and S* number of unicast incoming packets

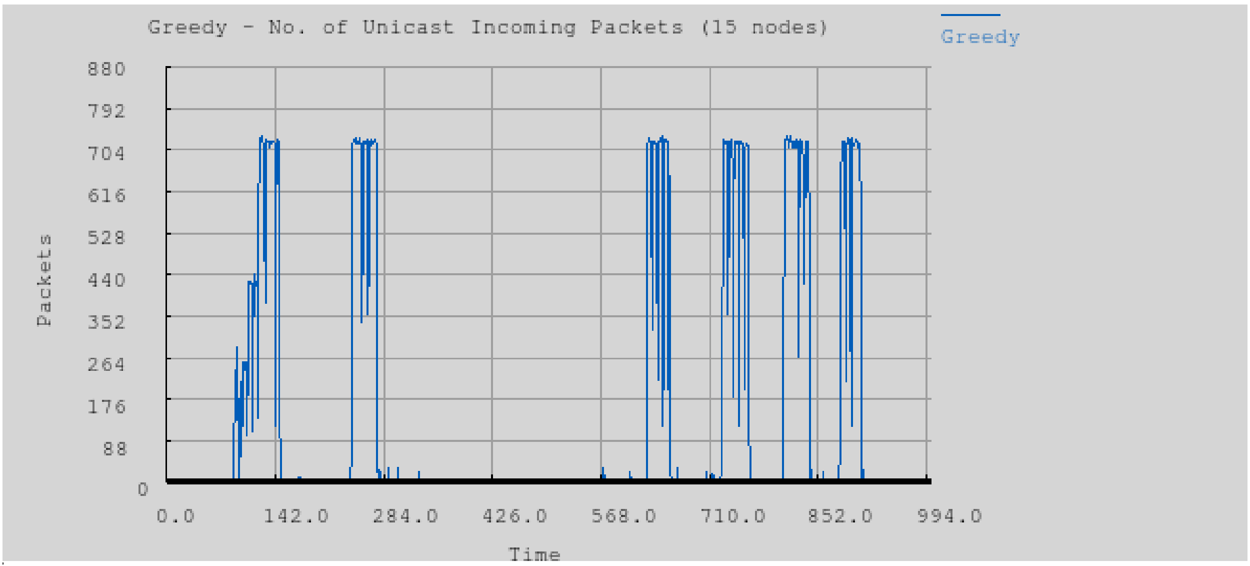


Figure 4.1 (a): No. of unicast incoming packets - Greedy (15 nodes)

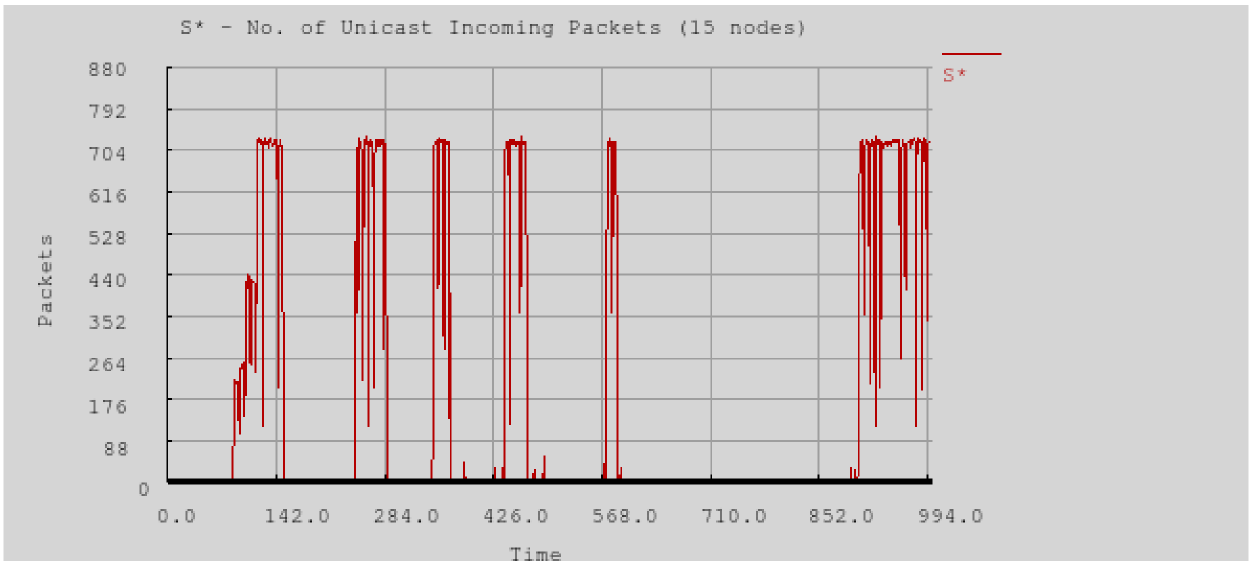


Figure 4.1 (b): No. of unicast incoming packets - S* (15 nodes)

Fig. 4.1 (a) and (b) depict performance over the network size of 15 nodes for Greedy and S* respectively. Both methods recorded moments of forwarding where the number of unicast packets delivered reached beyond 704 level on the vertical axis; although the number of S* graph columns appear to be slightly more on the horizontal axis. The two graphs further show patterns where some areas are comparatively congruent during the initial period of 0-284 seconds, while afterward there are complementarily alternating graph columns and empty spaces. It does appear that either of the two algorithms works best in certain regions of this simulation environment, and thus would fit different VANET scenarios best.

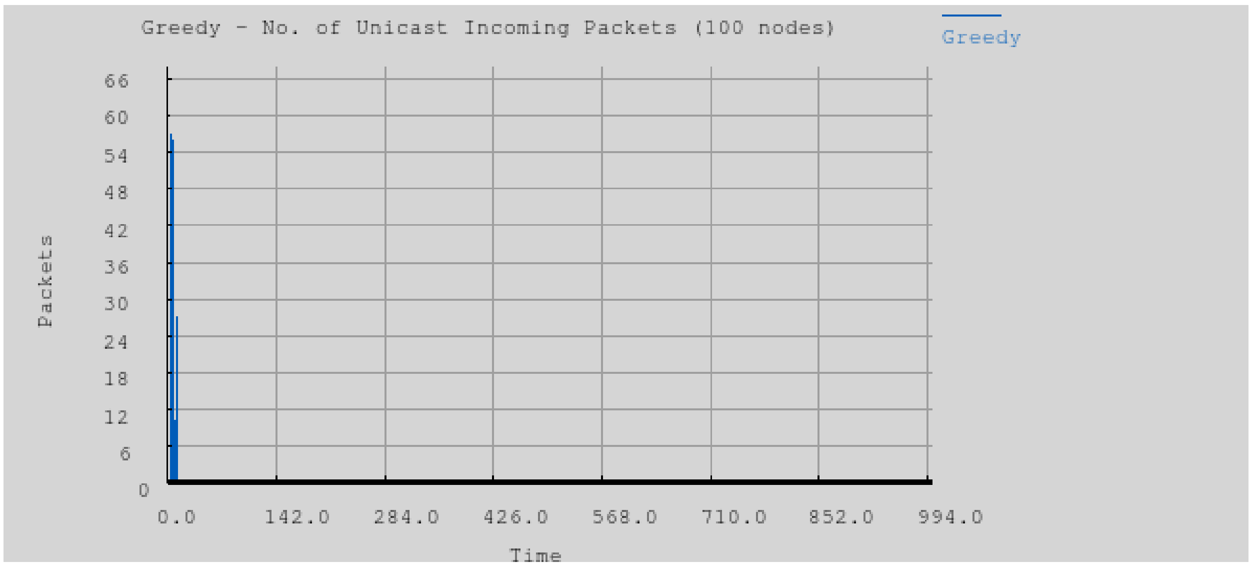


Figure 4.1 (c): No. of unicast incoming packets - Greedy (100 nodes)

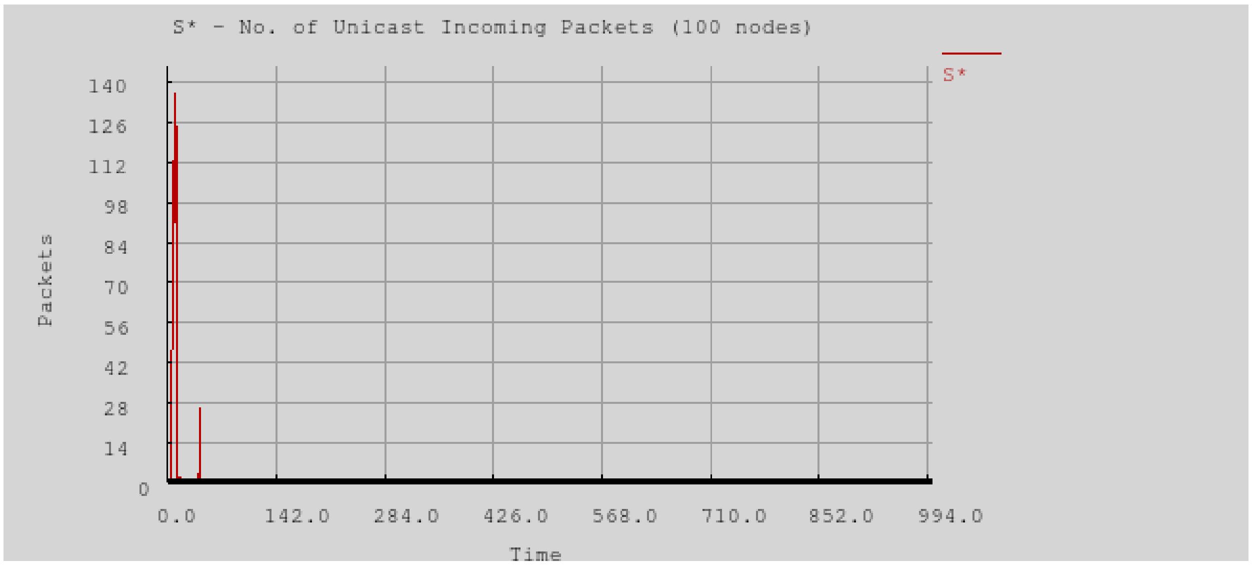


Figure 4.1 (d): No. of unicast incoming packets - S* (100 nodes)

Fig. 4.1 (c) and (d) show the graphs of incoming packets in the 100-node network. The network density of 1/540000m^2 is very sparse, which easily and quickly partitions, resulting also in sparse graphs. However both Greedy and S* forwarding effected some packets delivery at the commencement region; with S* rising to a level around 135 compared to Greedy's 56.

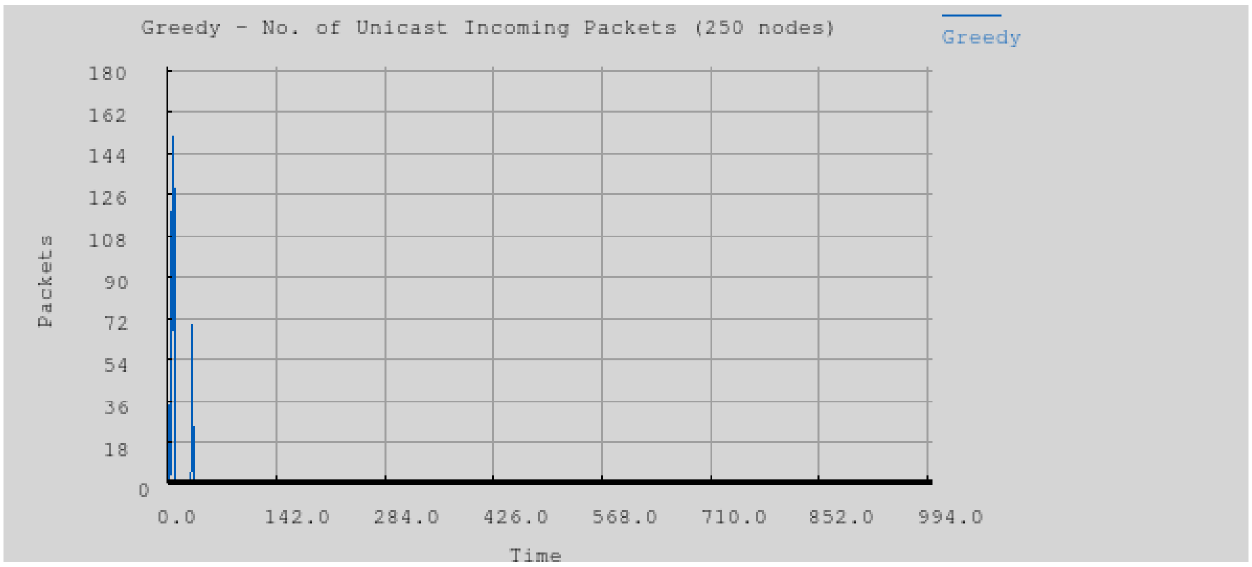


Figure 4.1 (e): No. of unicast incoming packets - Greedy (250 nodes)

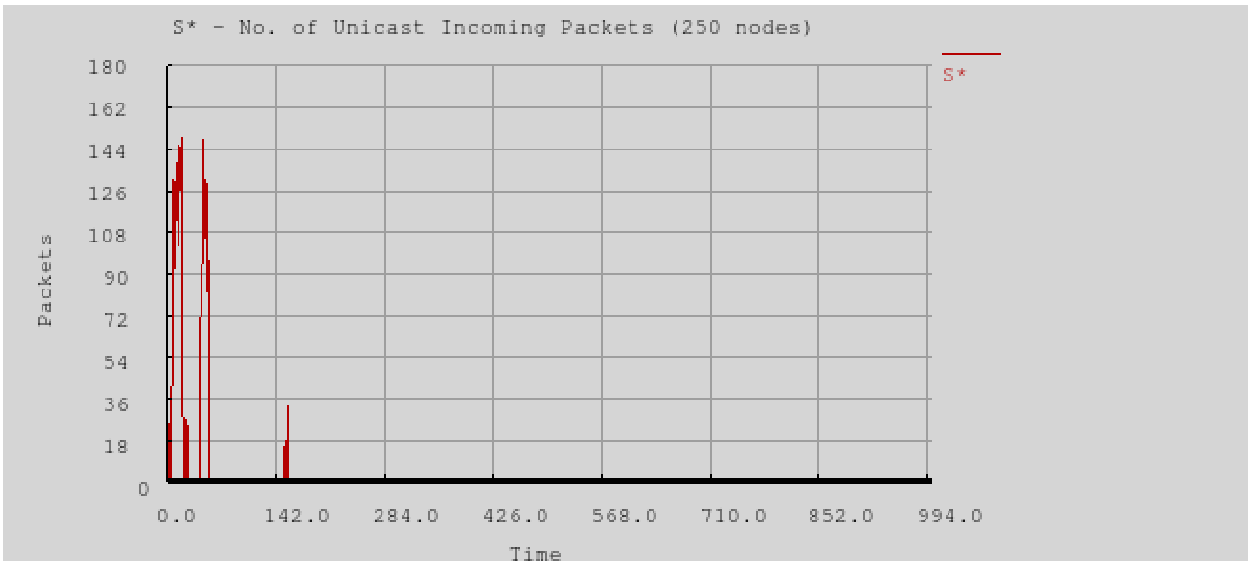


Figure 4.1 (f): No. of unicast incoming packets - S* (250 nodes)

Fig. 4.1 (e) and (f) show the graphs of incoming packets in the 250-node network. Again, successful packets delivery occurred at the commencement only. However, S* delivered a bit more packets showing two prominent graph columns that rises up to 144 level, compared to one column for Greedy.

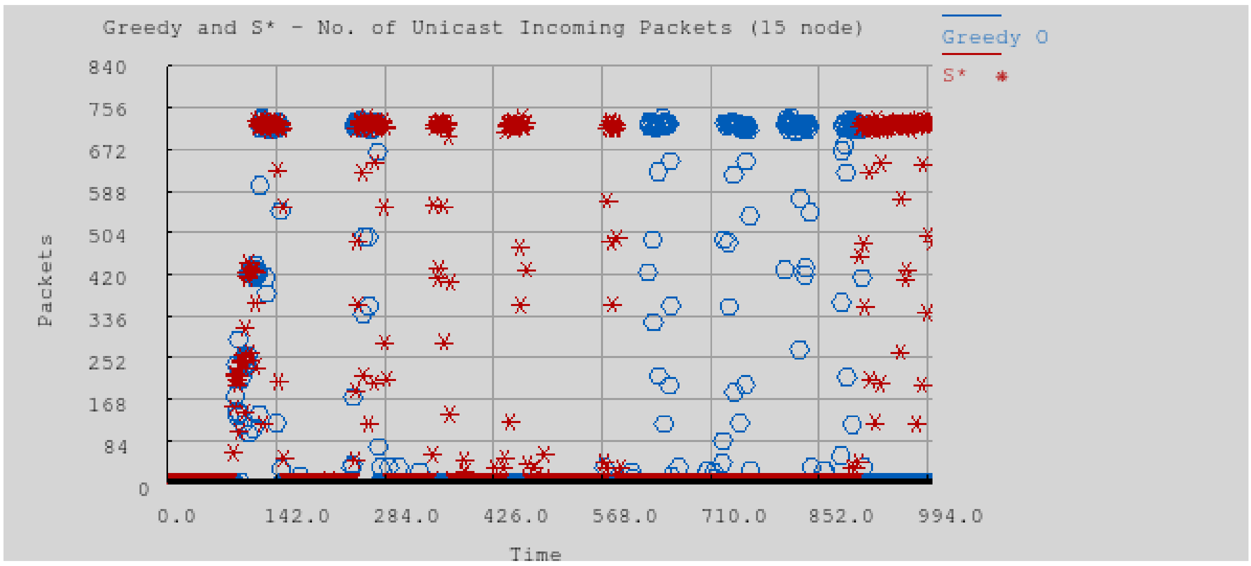


Fig. 4.1 (g): No. of unicast incoming packets - Greedy and S* (15 nodes)

Fig. 4.1 (g) shows the Greedy and the S* unicast incoming packets delivery rates in discrete values in the 15-node network, which reflects the superimposition of the graphs of Fig. 4.1 (a) & (b) relatively. During the period 0-284s both algorithms displayed activities; between 284-568s S* mostly demonstrated activity; between 568-852s Greedy mostly demonstrated activity; and between 852-994s S* mostly demonstrated activity. Therefore, judging from the foregoing alternating activity graphs of Greedy and S*, it appears that either of the algorithms would fit certain situations best in the VANET scenario.

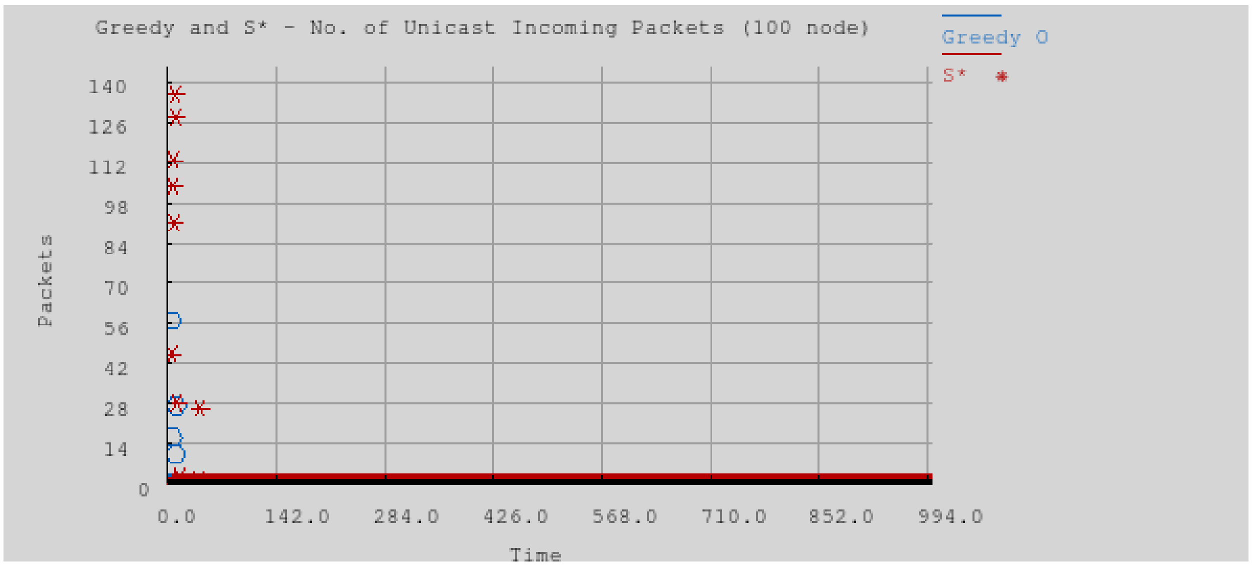


Fig. 4.1 (h): No. of unicast incoming packets - Greedy and S* (100 nodes)

Fig. 4.1 (h) shows the Greedy and the S* unicast incoming packets delivery rates in discrete values in the 100-node network, which reflects the superimposition of the graphs of 4.1 (c) & (d) relatively. S* has more delivery values than does Greedy in the sparse and soon partitioned network.

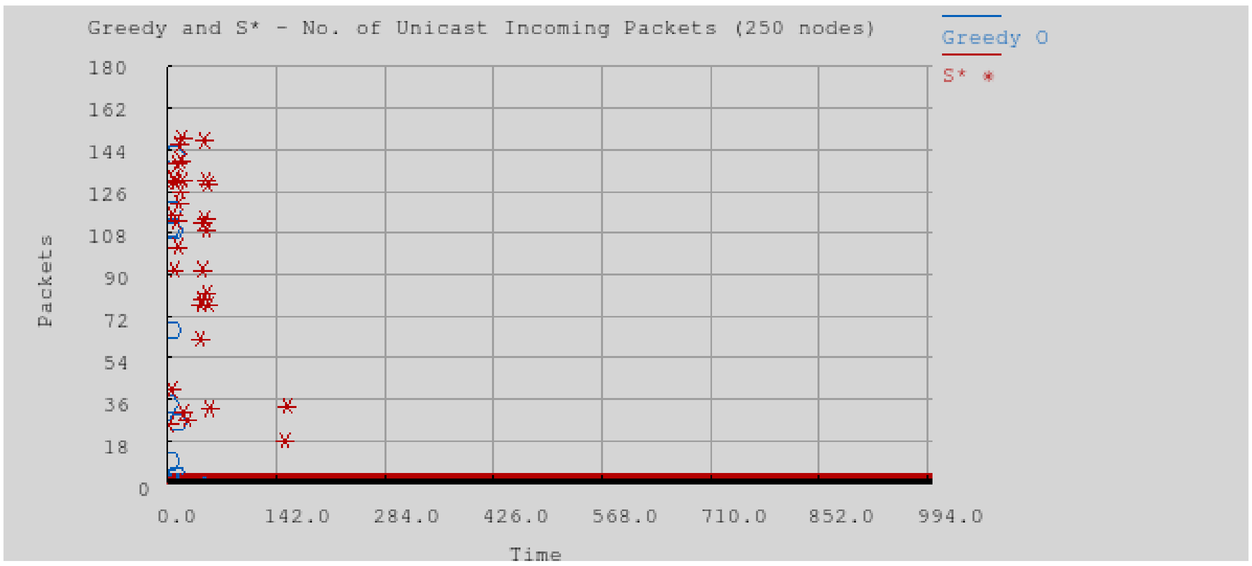


Fig. 4.1 (i): No. of unicast incoming packets - S* (250 nodes)

Fig. 4.1 (i) shows the Greedy and the S* unicast incoming packets delivery rates in discrete values in the 250-node network, which reflects the superimposition of the graphs of 4.1 (e) & (f) relatively. S* has more delivery values than does Greedy.

The graphs in Fig. 4.1 (a) - (i) show that S* tends to successfully deliver more packets than Greedy over the different density networks simulated. The better delivery rate could be attributable to the fact that the packets of S* travel over shorter distances in reaching the destination and therefore effect more throughput than that of Greedy.

4.1.2 Comparison of Greedy and S* unicast outgoing packets

Fig. 4.2 (a) - (i) reflect the number of unicast outgoing packets as dispatched from the source node over the network sizes shown and for the Greedy and the S* forwarding methods.

Figure 4.2 (a) - (i): Comparison of S* and Greedy number of unicast outgoing packets

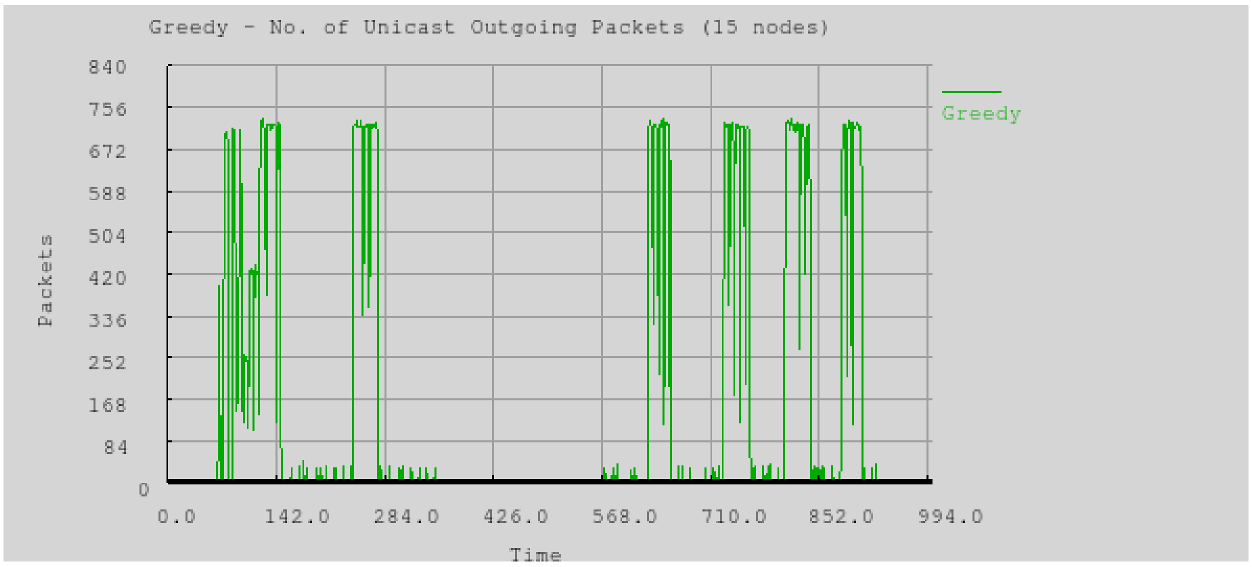


Figure 4.2 (a): No. of unicast outgoing packets - Greedy (15 nodes)

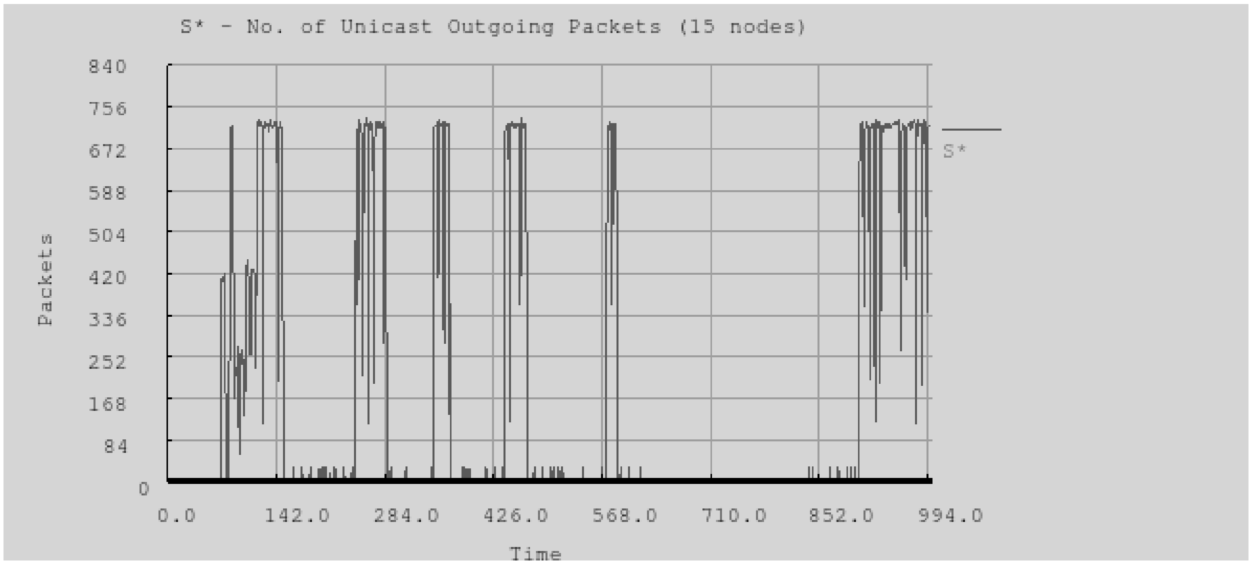


Figure 4.2 (b): No. of unicast outgoing packets - S* (15 nodes)

Fig. 4.2 (a) and (b) of 15-node size outgoing unicast graphs are similar in patterns to Fig. 4.1 (a) and (b) respectively; denoting that most of the packets successfully dispatched were also delivered by both the Greedy and the S* methods. However, the dissimilarities in the outgoing unicast graphs (Fig. 4.2 (a) and (b)) reflect the fact that the algorithms influence packets dispatching too, and differently.

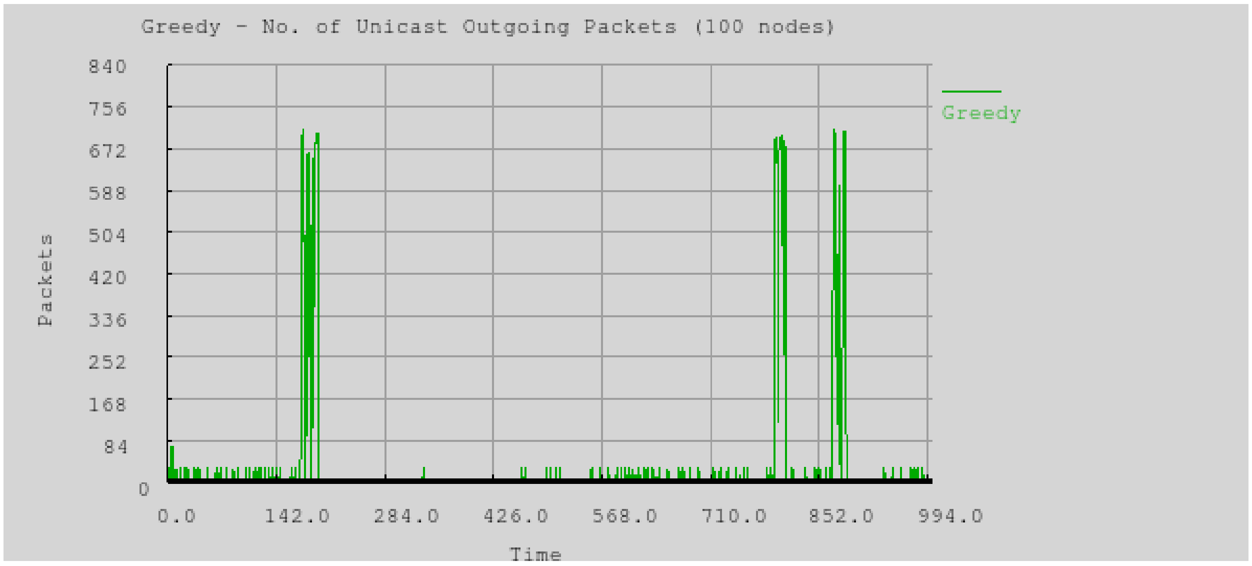


Figure 4.2 (c): No. of unicast outgoing packets - Greedy (100 nodes)

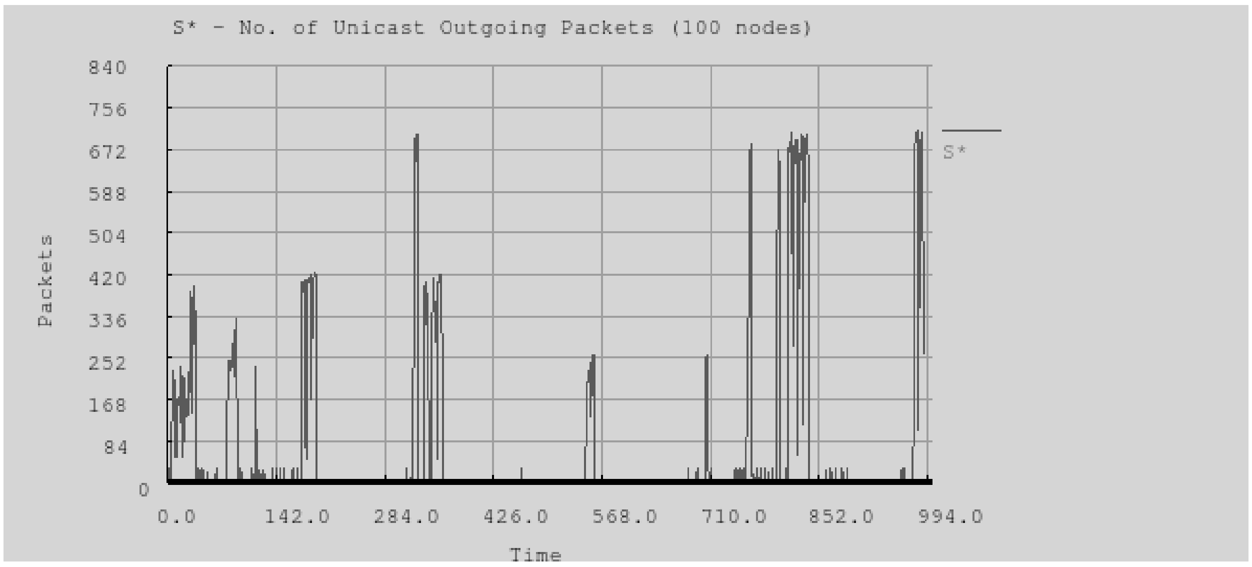


Figure 4.2 (d): No. of unicast outgoing packets - S* (100 nodes)

In the case of Fig. 4.2 (c) and (d), the outgoing dispatch of packets in the 100-node network size does not match the corresponding incoming packets' patterns that are seen in Fig. 4.1 (c) and (d), i.e. most packets dispatched were not delivered. The disconnection of links in the sparse network may have caused the mismatch in the outgoing to incoming packets ratios. Nevertheless, the levels of outgoing packets of S* surpasses that of Greedy. Both graphs have three principal graph columns that reached beyond 672; while S* in addition has some columns that reached beyond 420 and 252 levels in the number of packets delivered at some periods.

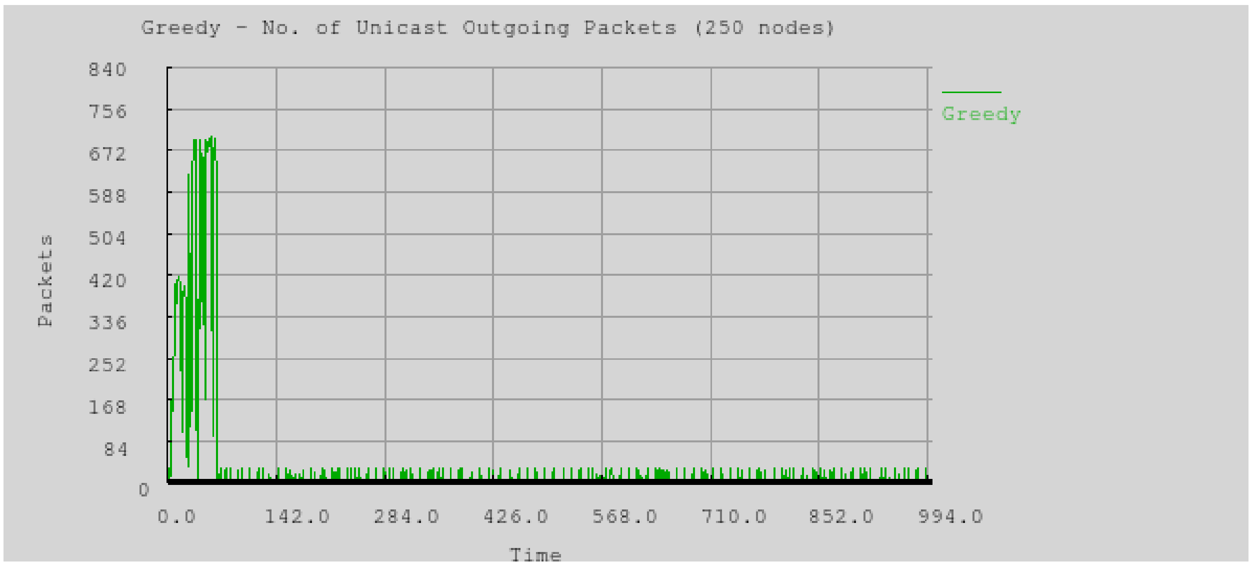


Figure 4.2 (e): No. of unicast outgoing packets - Greedy (250 nodes)

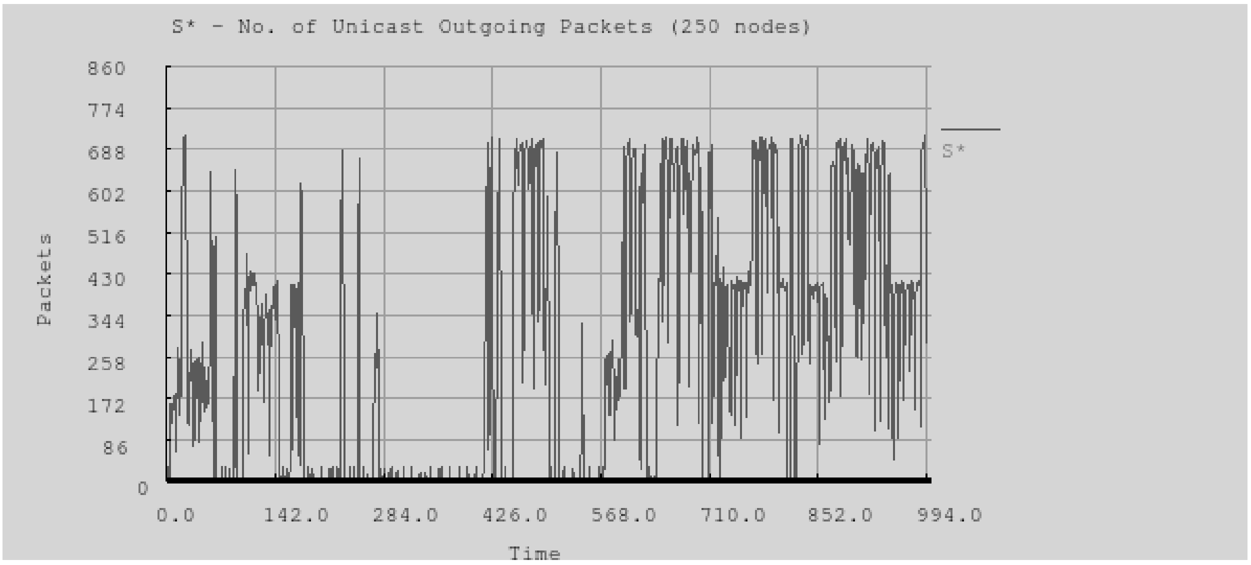


Figure 4.2 (f): No. of unicast outgoing packets - S* (250 nodes)

Fig. 4.2 (e) and (f) of 250-node network for the outgoing number of unicast packets' graphs show a far more pronounced number of columns for S* than for Greedy. The S* graph has a spread of columns that reaches beyond the 672 level; but Greedy has only one period of such column, with the rest being at low level around 25. S* clearly demonstrated better packets dispatch.

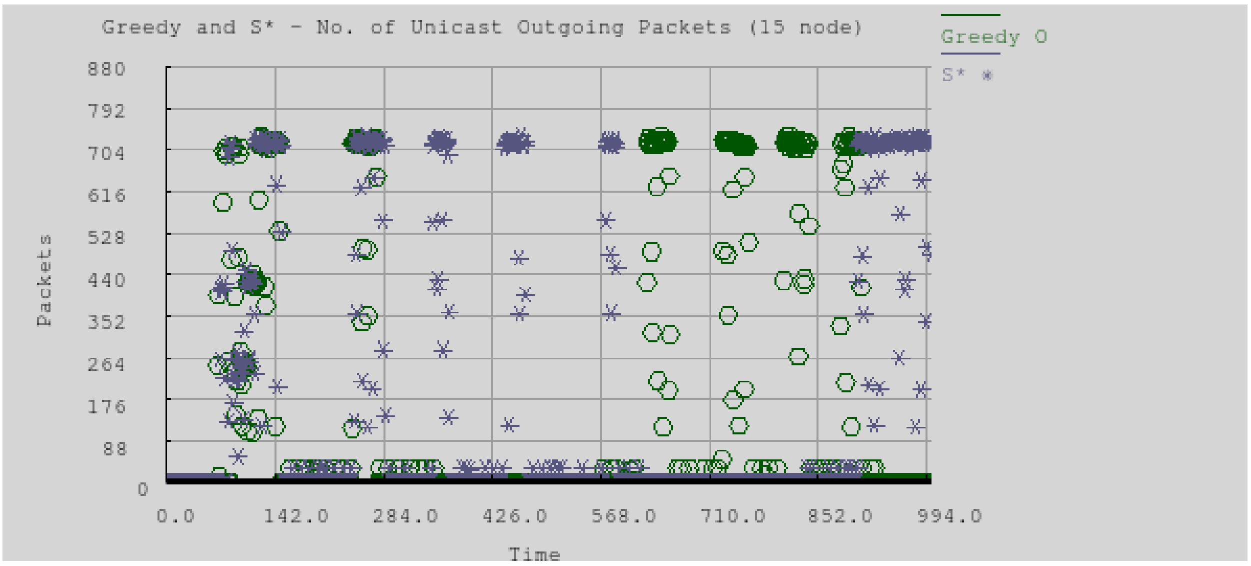


Fig. 4.2 (g): No. of unicast outgoing packets - Greedy and S* (15 nodes)

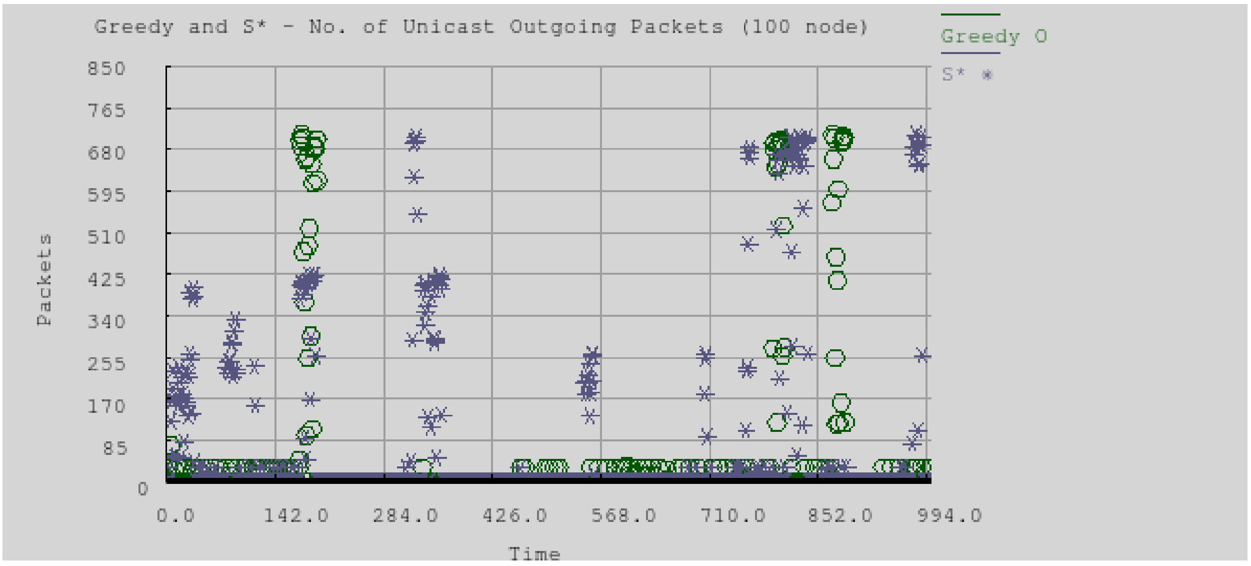


Fig. 4.2 (h): No. of unicast outgoing packets - Greedy and S* (100 nodes)

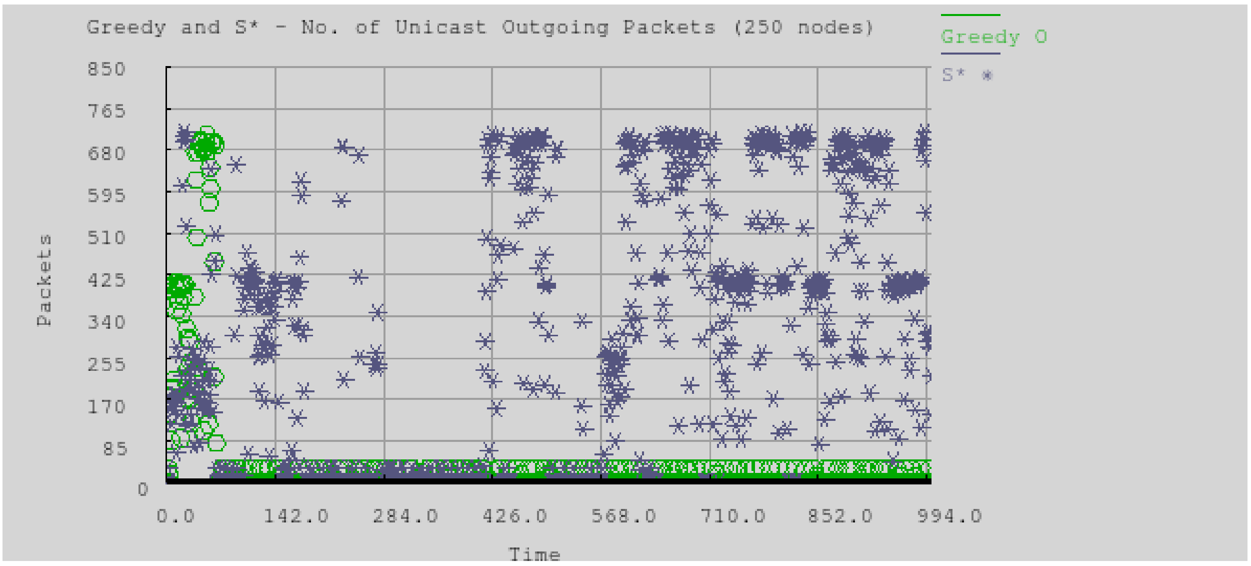


Fig. 4.2 (i): No. of unicast outgoing packets - Greedy and S* (250 nodes)

Fig. 4.2 (g) - (i) show Greedy and S* unicast outgoing packets delivery measures, which respectively reflect the superimpositions of graphs in Fig. 4.2 (a) & (b), 4.2 (c) & (d), and 4.2 (e) & (f). The S* packets dispatch rates appear higher than that of Greedy method in the three graphs.

In general the number of unicast incoming and outgoing graphs of packets delivery success rate demonstrate that S* performs better than the Greedy algorithm over the different density networks simulated.

4.2 Effect of the orthogonal distance to baseline search-space limiting

We review the simulation results of the vertical distance to baseline restriction (the *VDIST* or *VDIST_MAX*) performances in this section. The S* greedy forwarding method was used in the simulations that produced these results, and for a 250-node moderately dense network. We show results for both number of unicast incoming and outgoing packets delivery success rates; and for different neighbour node selection limits, including 20m, 80m, 250m, 500m and 900m. The results' graphs show that the varying of the orthogonal extent to the baseline area from which the set of potential next-hop nodes are selected does affect packets delivery success rate in VANET.

4.2.1 Effect of vertical distance selection limit of nodes on unicast incoming packets

Fig. 4.3 (a) - (h) reflect the number of unicast incoming packets received at the destination node over the network size of 250 nodes for regulated vertical distances *VDIST_MAX* of 20m, 80m, 250m, 500m and 900m respectively.

Figure 4.3 (a) - (e): The effect of Vdist nodes selection limits on unicast incoming packets

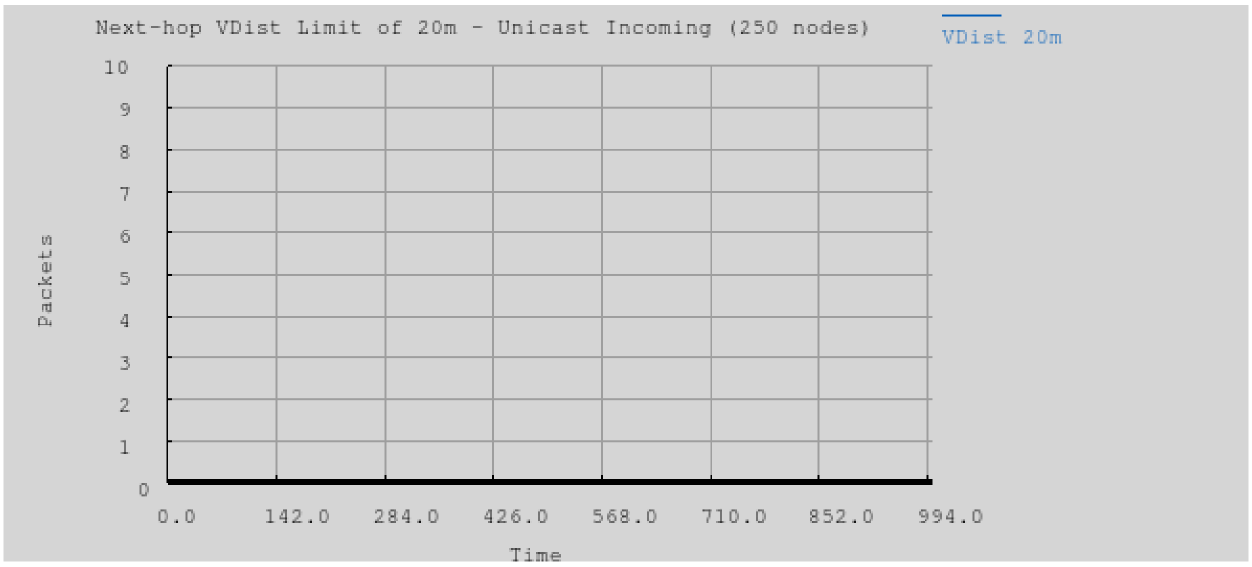


Fig. 4.3 (a): No. of unicast incoming packets delivery over Vdist limit of 20m (an empty graph)

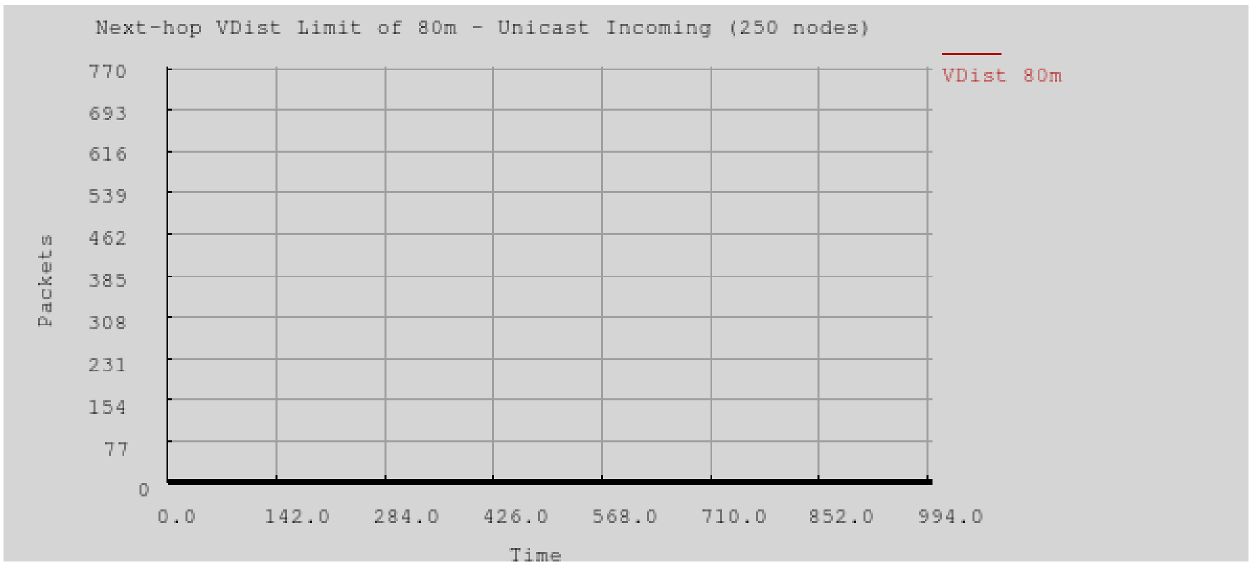


Fig. 4.3 (b): No. of unicast incoming packets delivery over Vdist limit of 80m (an empty graph)

Fig. 4.3 (a) and (b) show nil graphs for the number of unicast incoming packets delivered for *VDIST_MAX* setting of 20m and 80m. The 20m and 80m limit areas may have been too narrow as to encompass pertinent qualified nodes in these cases; hence effective packets forwarding failed to take place.

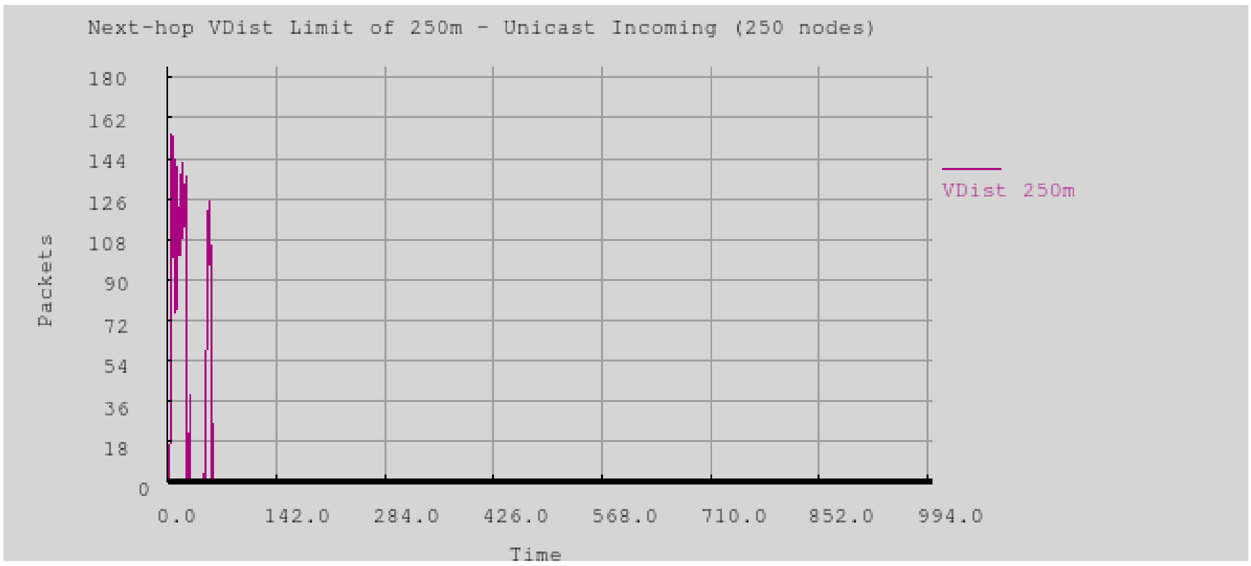


Fig. 4.3 (c): No. of unicast incoming packets delivery over Vdist limit of 250m

Fig 4.3 (c) reflect some packets delivery value that reached 144 level during the initial forwarding period of around 0-50s. In comparison to Fig. 4.3 (a) & (b) graphs that are empty for the 20m and 80m Vdist restrictions, the 250m restriction limit enabled some nodes to forward packets in the 1/216000m^2 density network. It implies that Vdist restriction value must be somewhere above 80m in this network for effective packets forwarding to take place.

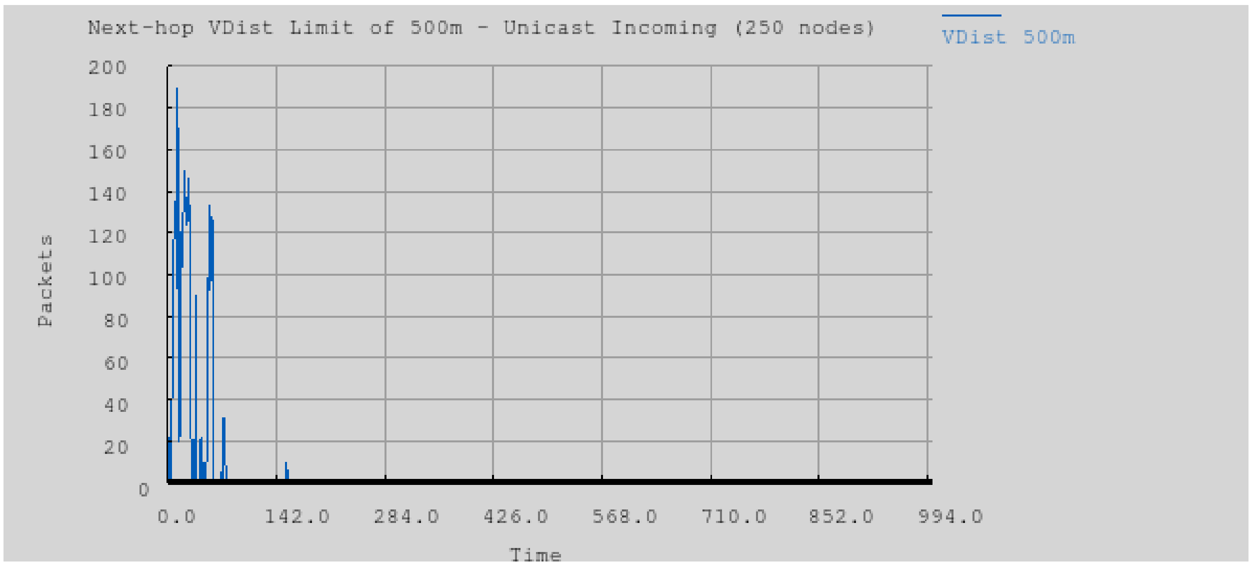


Fig. 4.3 (d): No. of unicast incoming packets delivery over Vdist limit of 500m

In comparison to Fig 4.3 (c) of 250m restriction, Fig 4.3 (d) of 500m *VDIST_MAX* shows increased packets delivery that is up to the 190 level and for a period longer than 0-50s. The difference between the two graphs primarily reflects the effectiveness of the VDIST restriction function. The packets delivery level at 250m is lower than that of 500m Vdist, which implies that in the latter case more nodes were provided for evaluation as next-hop, with better selection and forwarding opportunities.

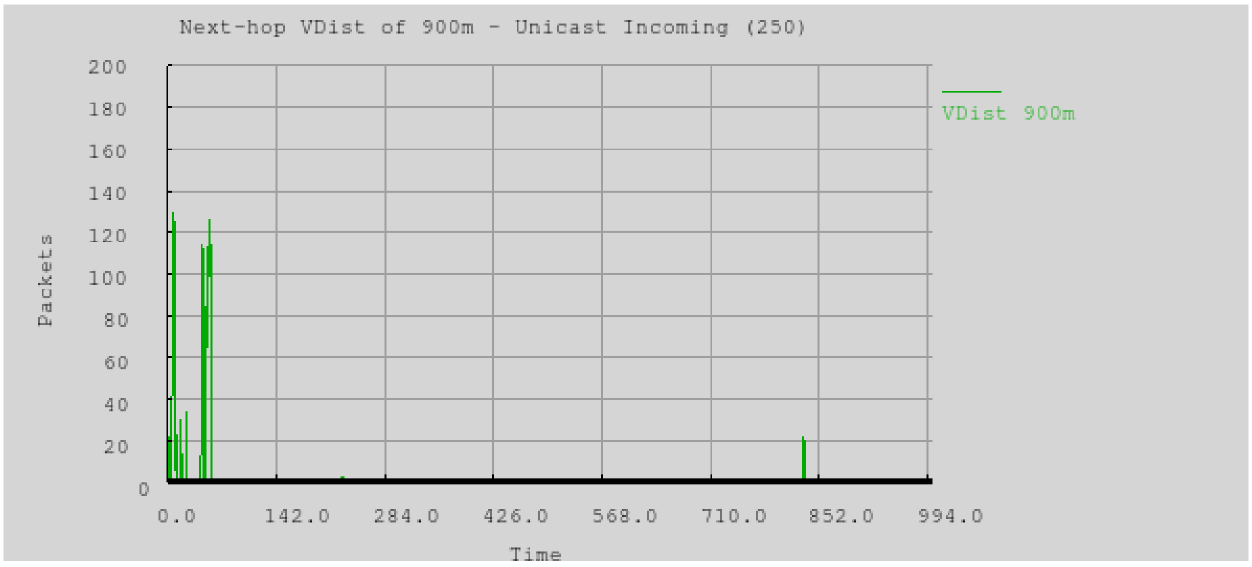


Fig. 4.3 (e): No. of unicast incoming packets delivery over Vdist limit of 900m

Fig. 4.3 (e) reflects packet delivery at 900m Vdist restriction; which is slightly above the full transmission range of 885m. The packets delivery level is at about 125, a value that is much lower than the 190 of 500m Vdist in Fig. 4.3 (d) with a graph that also spans a longer period. Both graphs display a short strand of packets delivery activity at some later period respectively.

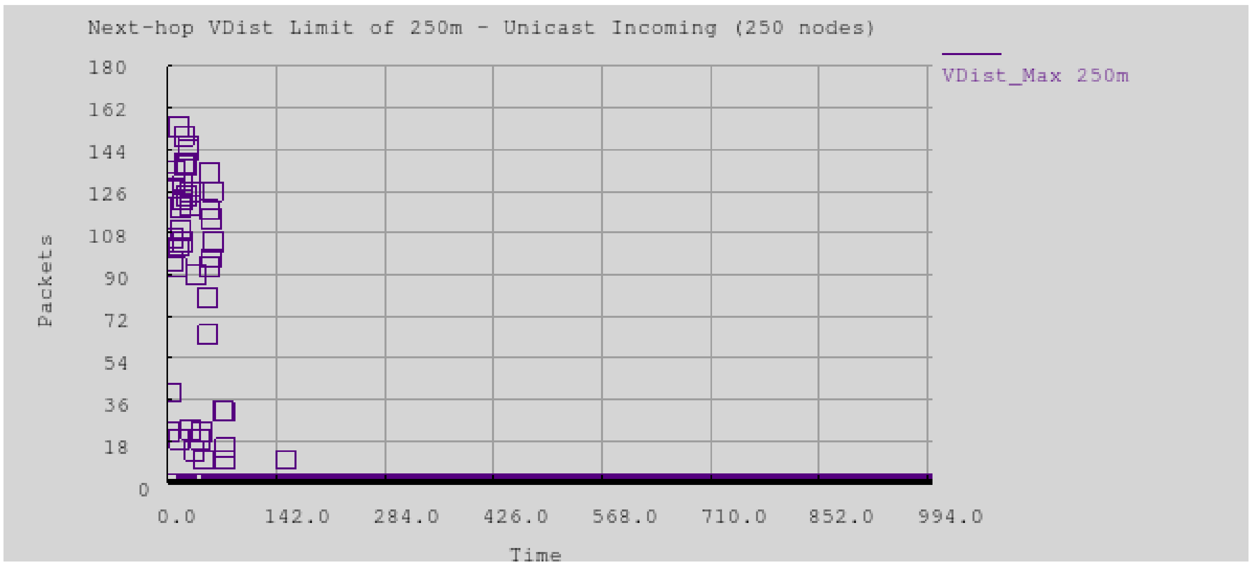


Fig. 4.3 (f): No. of unicast incoming packets delivery over Vdist limit of 250m

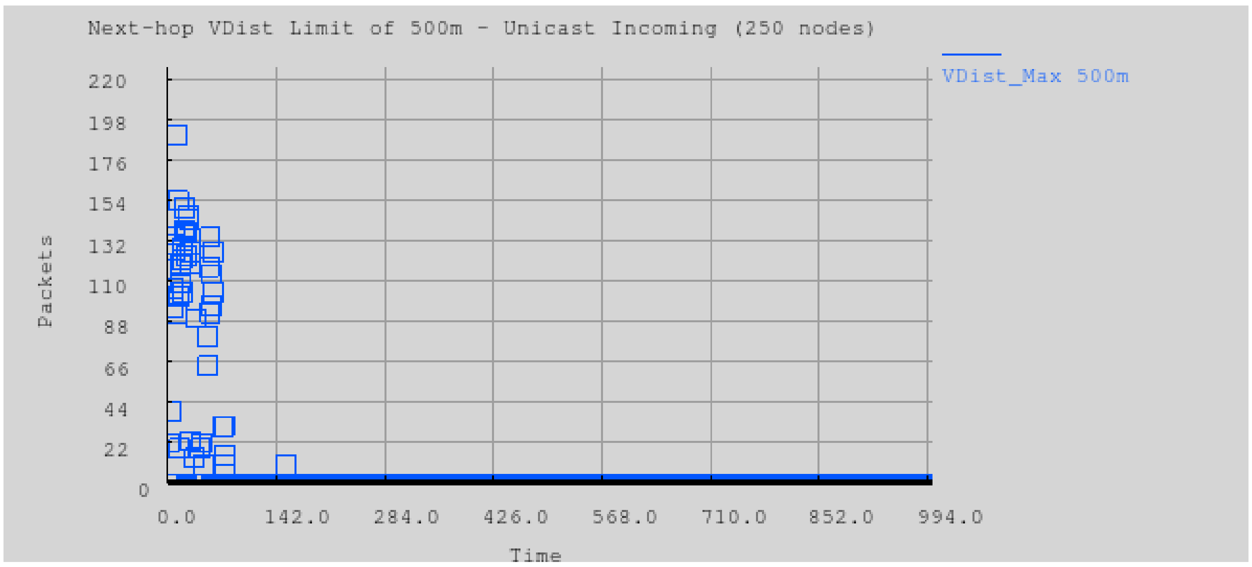


Fig. 4.3 (g): No. of unicast incoming packets delivery over Vdist limit of 500m

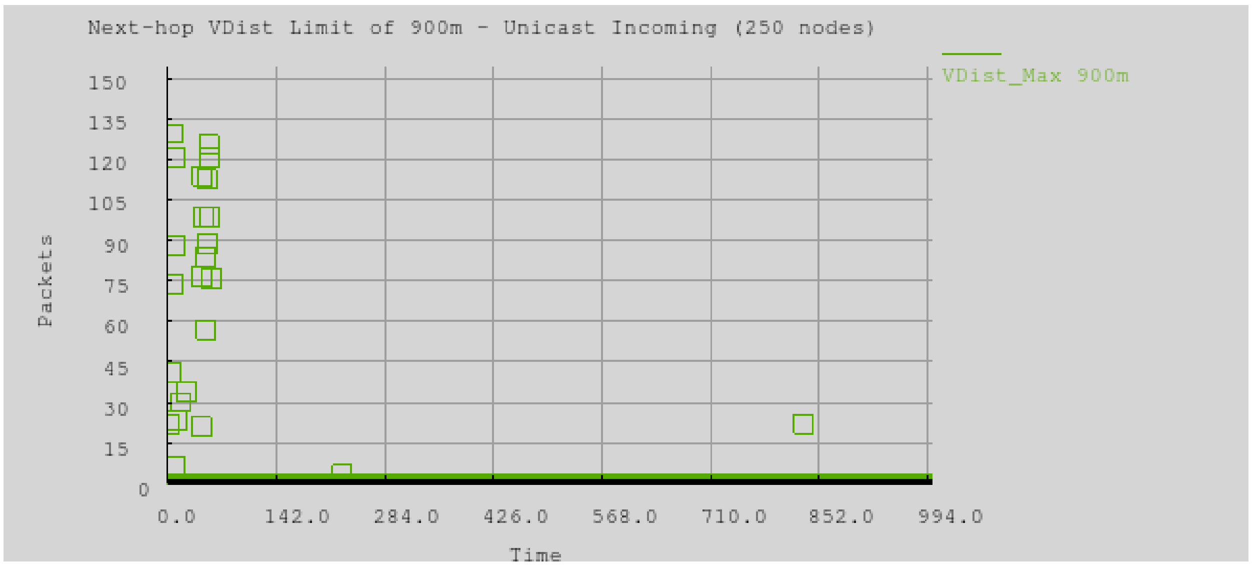


Fig. 4.3 (h): No. of unicast incoming packets delivery over Vdist limit of 900m

Fig. 4.3 (f) - (h) show the unicast incoming packets delivery graphs in discrete values, which correspond to the graphs in Fig. 4.3 (c) - (e) respectively. The graphs of *VDIST_MAX* at 250m in Fig. 4.3 (f) and at 500m in Fig. 4.3 (g) are almost similar, but the latter's graph shows delivery value at a level closer to the 198 mark while the former reaches only close to 162. The comparison of the graphs of Fig. 4.3 (g) and Fig. 4.3 (h) again show the better performance of the 500m over the 900m *VDIST_MAX* selection.

4.2.2 Effect of vertical distance selection limit of nodes on unicast outgoing packets

Fig. 4.4 (a) - (e) reflect the number of unicast outgoing packets sent from the source node over the network size of 250 nodes for regulated vertical distances *VDIST_MAX* of 20m, 80m, 250m, 500m and 900m respectively.

Figure 4.4 (a) - (e): The effect of Vdist nodes selection limits on unicast outgoing packets

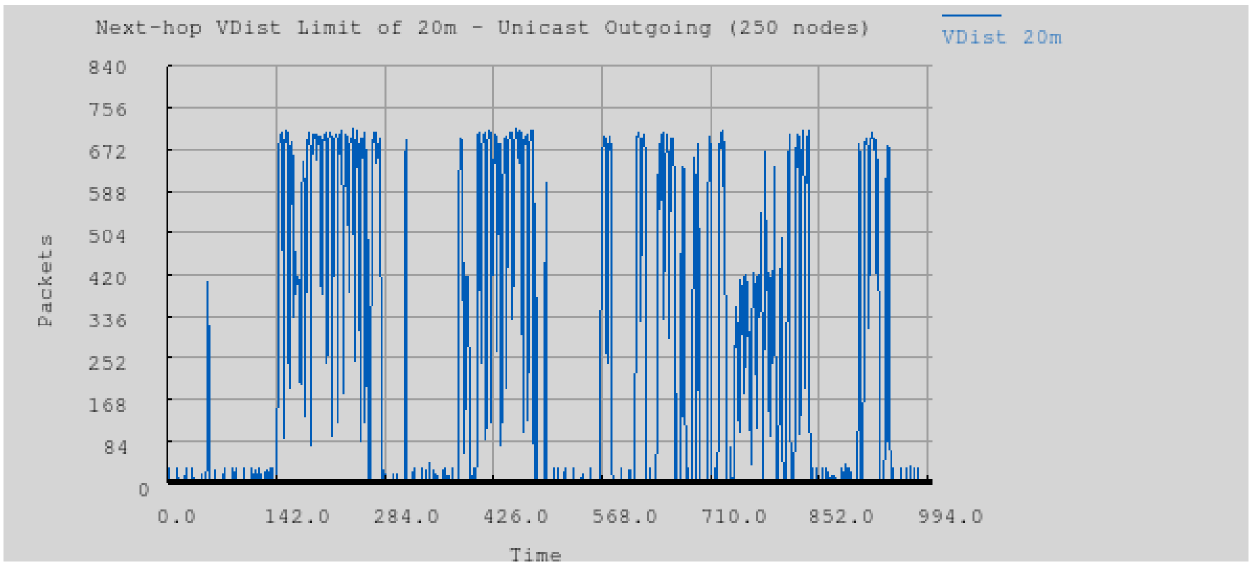


Fig. 4.4 (a): No. of unicast outgoing packets delivery with Vdist limit of 20m

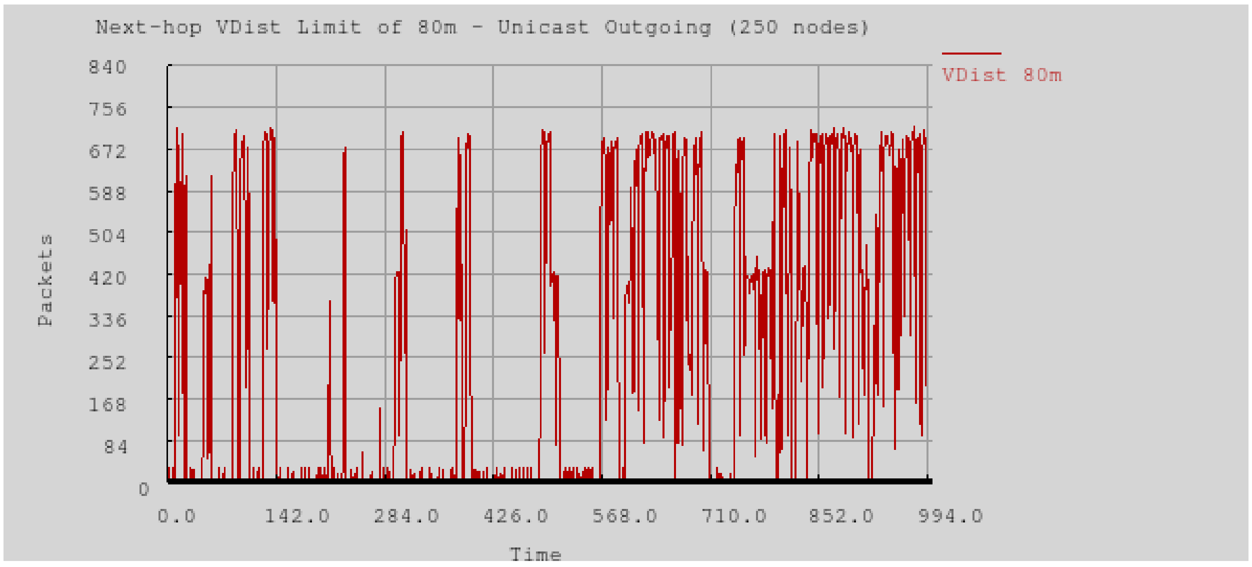


Fig. 4.4 (b): No. of unicast outgoing packets delivery with Vdist limit of 80m

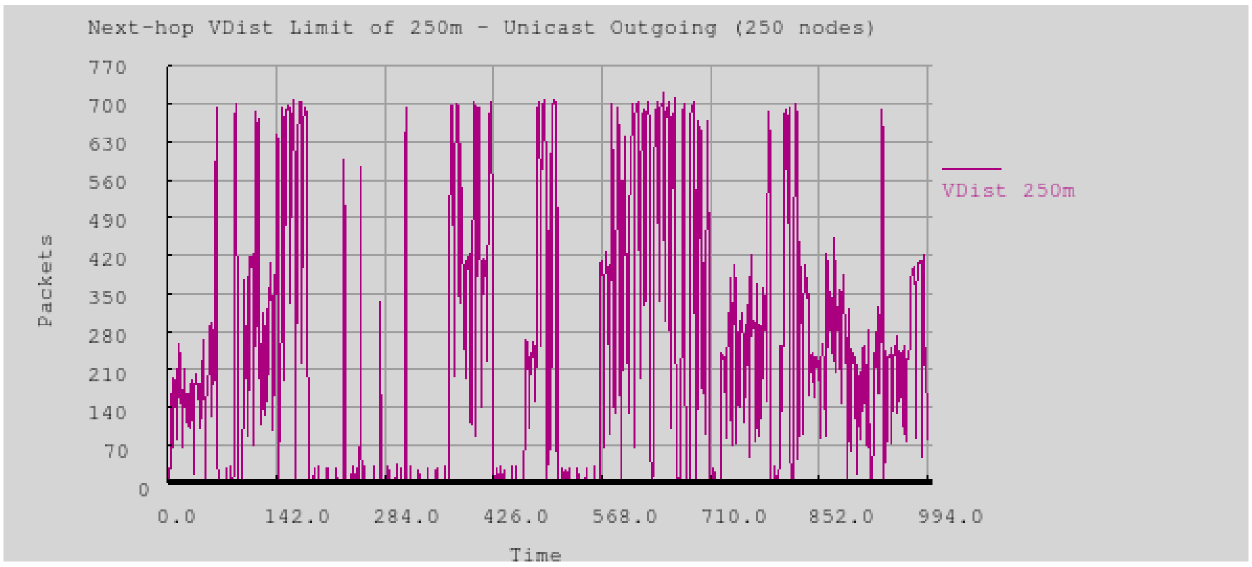


Fig. 4.4 (c): No. of unicast outgoing packets delivery with Vdist limit of 250m

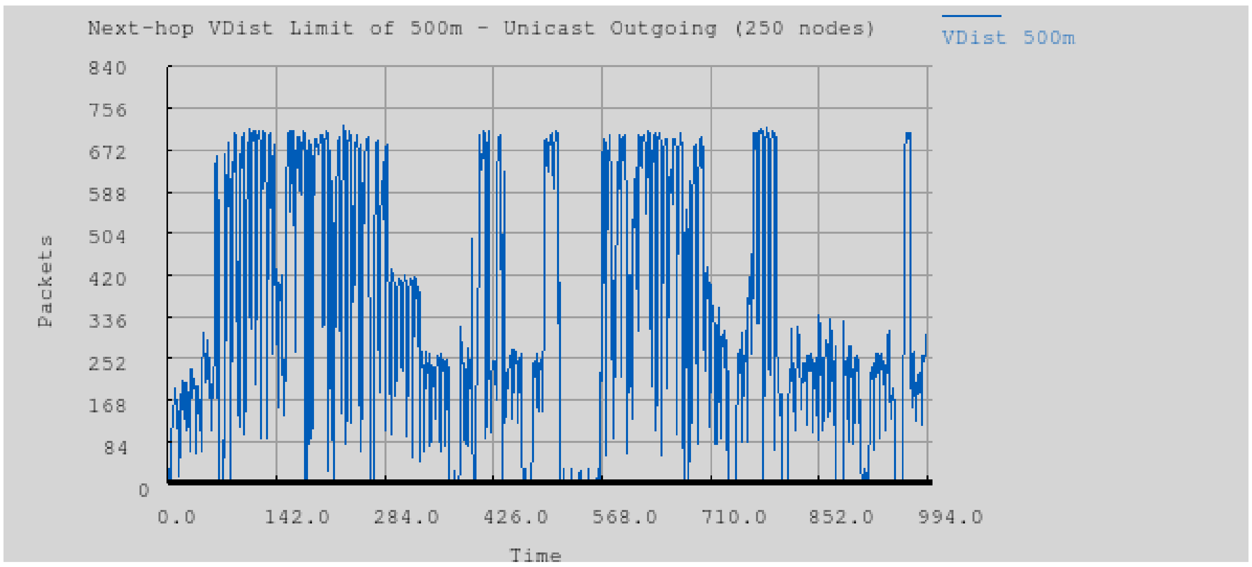


Fig. 4.4 (d): No. of unicast outgoing packets delivery with Vdist limit of 500m

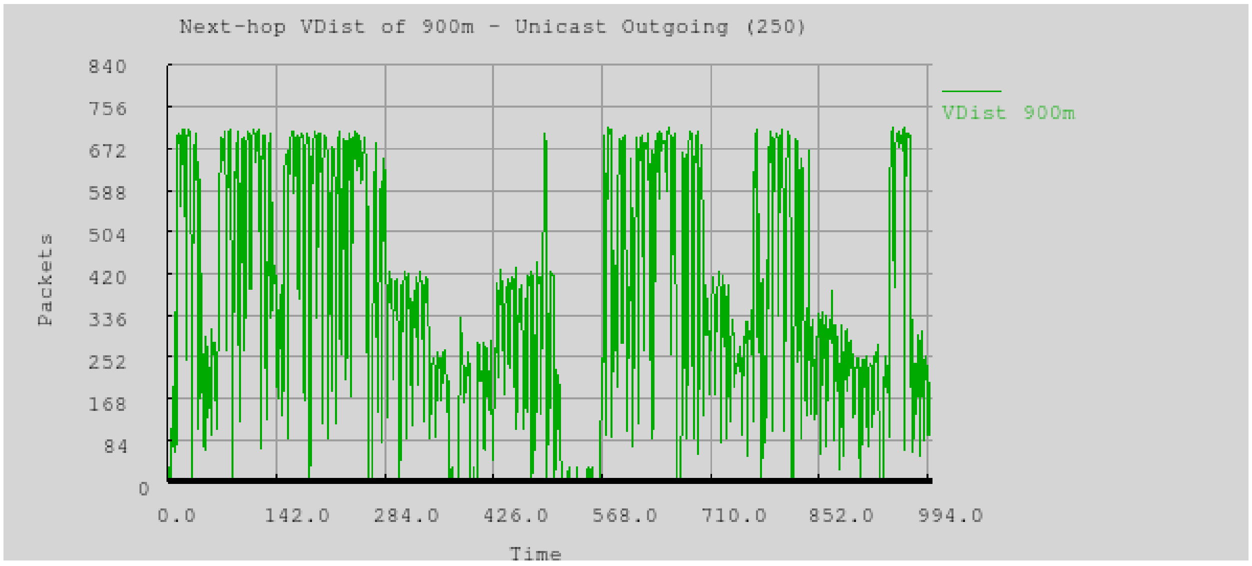


Fig. 4.4 (e): No. of unicast outgoing packets delivery with Vdist limit of 900m

The outgoing unicast graphs of Vdist of 20m, 80m, 250m, 500m and 900m, in the foregoing Fig. 4.4 (a) - (e) all reflect packets dispatch rate that reach the level of 700; and in general the performances are similar notwithstanding the different graph patterns. Furthermore, this form of graph patterns still appears, albeit with varied columnar proportions, when the relevant simulation files are re-run. The near-uniformity that the graphs display apparently implies that the *VDIST_MAX* limiting does not influence the unicast outgoing packets dispatching at the source nodes.

The near-uniformity of the unicast outgoing graphs of Fig. 4.4 (a) - (e) contrasts with those of unicast incoming graphs of Fig. 4.3 (a) - (e) that show dissimilarities for the different *VDIST_MAX* cases simulated. Fig. 4.2 (a) - (f) also reflect dissimilarities in the graphs of the unicast outgoing packets delivery rates. Therefore, packets dispatch rate from the source nodes can actually be influenced by changing network condition, although this is not the case with Fig. 4.4 (a) - (e) graph outcomes.

In conclusion, these results show that *VDIST_MAX* variations do not affect the number of unicast outgoing packets delivery rate, although it affects the unicast incoming rates.

4.3 Position-aware forwarding dynamics

Fig. 4.5 (a) - (d) and 4.6 (a) & (b) show GUI animation snapshots of packets forwarding simulation for the comparison of the table-driven method, using AODV [11], and the geographic greedy routing method; with packets hopping from the source node 12 to the destination node 7 in the VANET scenario. Table-driven methods' packets forwarding follow paths of pre-determined sequence of node linkages; while in geographic greedy methods, a next-hop is decided as a node closest to the destination among those within forwarding range.

Figure 4.5 (a) - (d): Snapshots of animated AODV packets forwarding

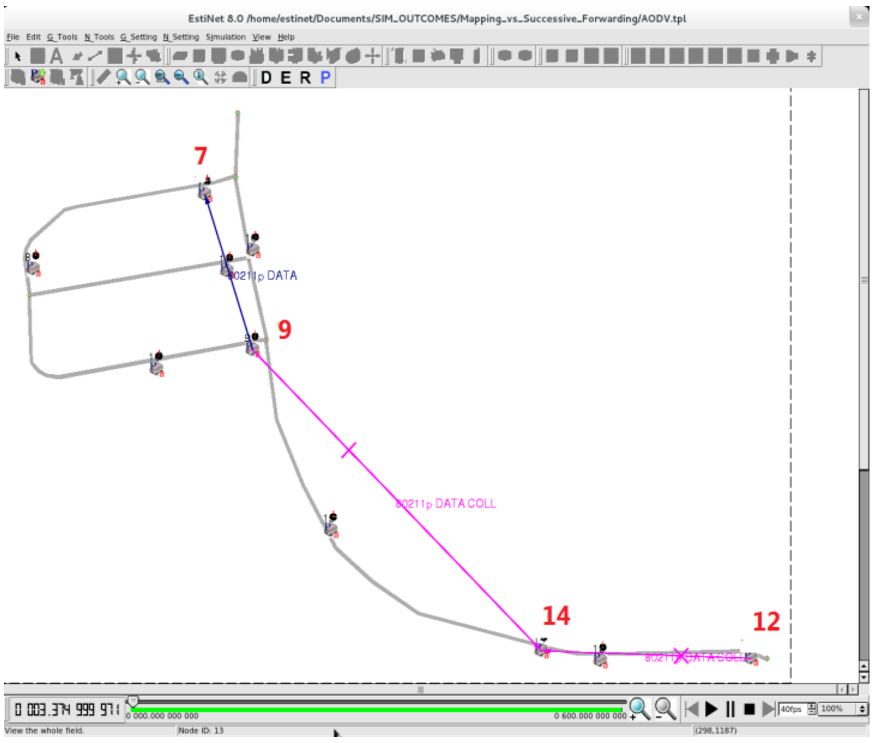

EstiNet 8.0 /home/estinet/Documents/SIM_OUTCOMES/Mapping_vs_Successive_Forwarding/AODV.tpl
File Edit G_Tools N_Tools G_Setting N_Setting Simulation View Help
D E R P
7
9
14
12
80211p DATA
80211p DATA COLL
0 003.374 999 971
0 000.000 000 000
0 600.000 000 000
40fps
100%
View the whole field.
Node ID: 13
(298,1187)

Figure 4.5 (a): An AODV forwarding instance – (1)

Fig. 4.5 (a) shows the beginning of the forwarding session for the table-driven AODV method after route discovery, from the lower right corner to the upper left corner as 12–14–9–7 sequence, which is in line with the current positional order of the nodes. Thereafter the mobile nodes moved about in the network arbitrarily, with switching of positions, breakage of links and network partitioning simultaneously taking place.

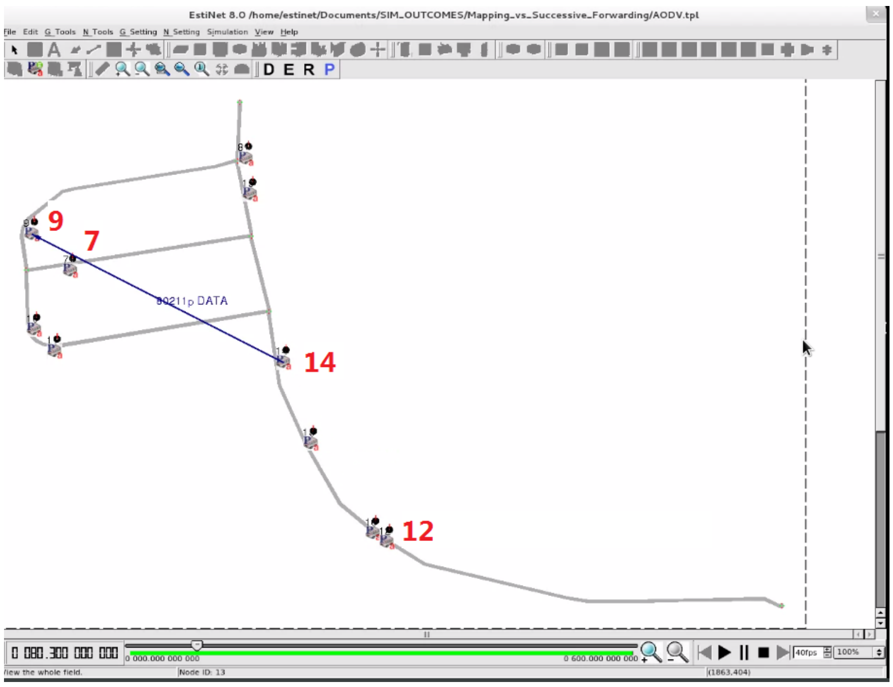

EstiNet 8.0 /home/estinet/Documents/SIM_OUTCOMES/Mapping_vs_Successive_Forwarding/AODV.tpl
9
7
802.11p DATA
14
12
D E R P

Figure 4.5 (b): An AODV forwarding instance – (2)

Nevertheless, the table-driven method kept firm to its initially derived forwarding sequence of 12–14–9–7, notwithstanding the transformation of the network that is taking place. In Fig. 4.5 (b), the packets coming from node 14 bypassed the destination node 7 and hopped over to node 9 from where the packets reroute back to node 7. It would have been expected that Node 14 forwards the packets directly to the now closer node 7.

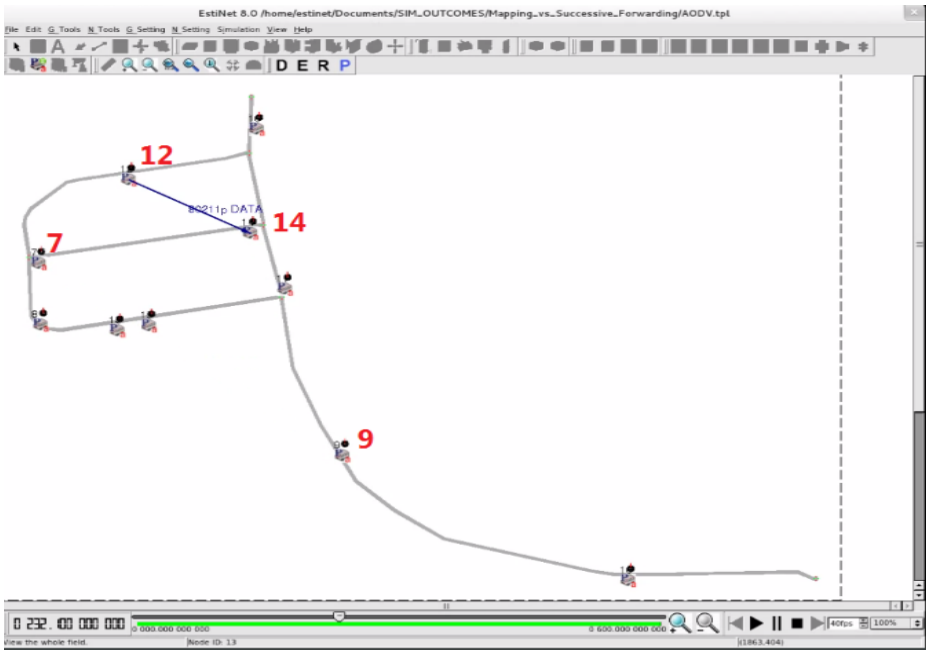

EstiNet 8.0 /home/estinet/Documents/SIM_OUTCOMES/Mapping_vs_Successive_Forwarding/AODV.tpl
12
14
7
9
802.11p DATA
0 232.400 000 000

Figure 4.5 (c): An AODV forwarding instance – (3)

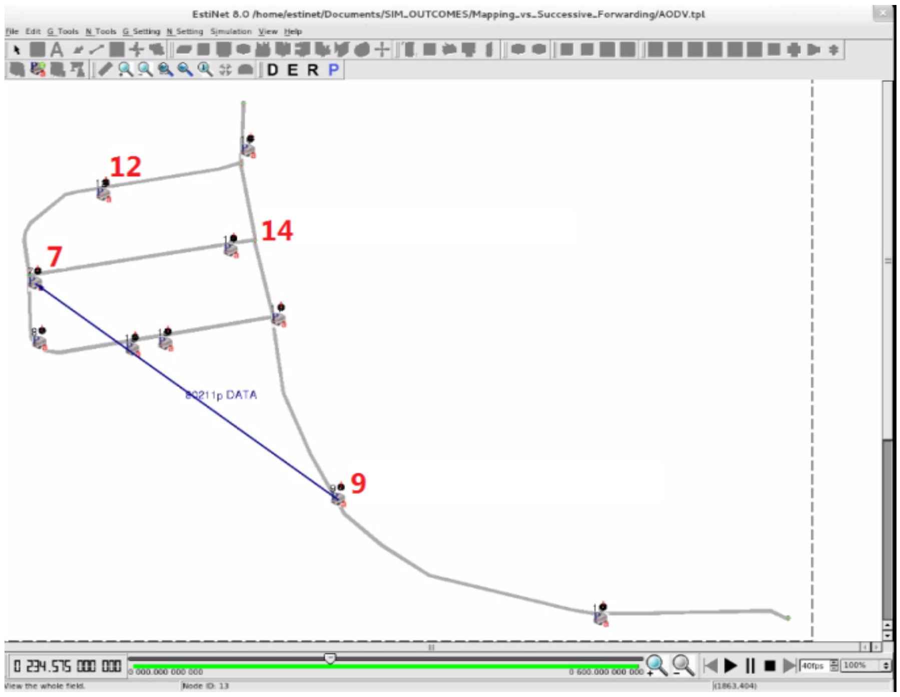

EstiNet 8.0 /home/estinet/Documents/SIM_OUTCOMES/Mapping_vs_Successive_Forwarding/AODV.tpl
12
14
7
9
802.11p DATA
0 234.575 000 000

Figure 4.5 (d): An AODV forwarding instance – (4)

Fig. 4.5 (c) and (d) show more instances of the forwarding anomaly along this inflexible routing sequence of the table-driven method, as packets are still routed round along 12–14–9–7 path, even when the source and destination nodes 12 and 7 are closest to each other. Thus, the table-driven method is not position-aware.

In comparison and in contrast to the foregoing, Fig. 4.6 (a) & (b) below demonstrates the position-aware packets forwarding method of a geographic greedy algorithm, which we simulated within the same VANET environment as the above.

Figure 4.6 (a) & (b): Snapshots of animated greedy packets forwarding as follows

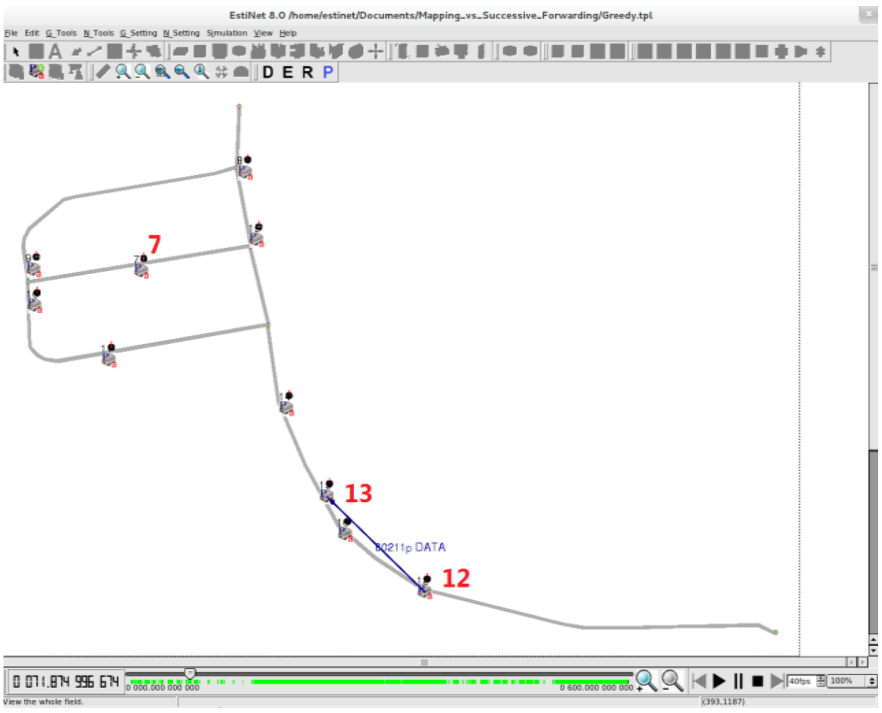

EstiNet 8.0 /home/estinet/Documents/Mapping_vs_Successive_Forwarding/Greedy.tpl
File Edit G_Tools N_Tools G_Setting N_Setting Simulation View Help
D E R P
7
13
80211p DATA
12
View the whole field.

Figure 4.6 (a): A location-aware greedy algorithm forwarding instance – (1)

Fig. 4.6a shows a snapshot commencement situation where the geographic greedy algorithm forwards packets from source node 12 to the next-hop node 13, which subsequently hops onward to the destination node 7.

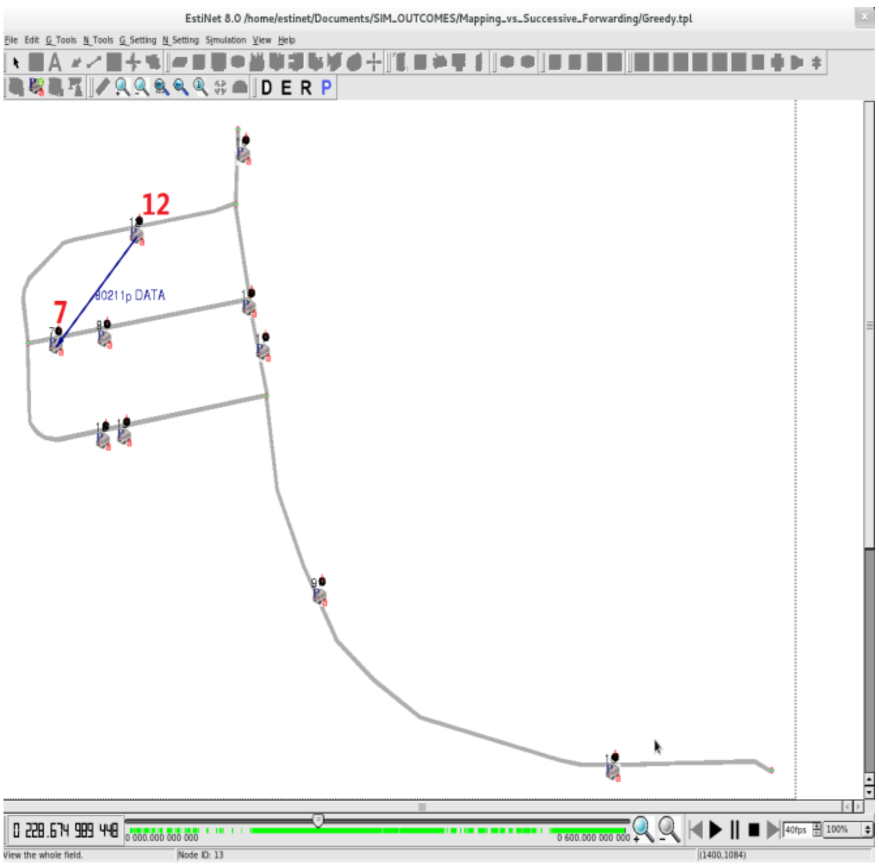

EstiNet 8.0 /home/estinet/Documents/SIM_OUTCOMES/Mapping_vs_Successive_Forwarding/Greedy.tpl
12
7
80211p DATA

Figure 4.6 (b): A location-aware greedy algorithm forwarding instance – (2)

The position-aware routing method makes appropriate changes in the choice of intermediate hopping nodes as the network configuration changes. In Fig. 4.6 (b), node 12 now forwards packets directly to node 7 in recognition of its currently close position. The greedy algorithm is immediately responsive with efficient forwarding to next-hop in VANET.

5.0 DISCUSSION & FUTURE WORK

In this section we discuss the S* next-hop selection technique as well as the DIST mechanism's efficacy in VANET; and we describe some possible enhancements. We also describe some future work prospects.

5.1 Efficiency of the S* forwarding technique

Overall, the S* method consistently delivered more packets than the basic Greedy method, as revealed in the outcomes of the simulation results presented in section 4.0. Although the S* packets forwarding technique performed functionally similar to the basic Greedy method in VANET, as both algorithms are closely related; however, the S* algorithm definitely has an influence on packets forwarding that has appeared better than that of the Greedy method. The incoming unicast packets delivery rate graphs reflect S* as performing better than Greedy in all cases evaluated. Moreover, S* significantly influence the unicast outgoing packets dispatches more than Greedy.

Although the Greedy next-hop selection technique has not been ratified for VANET geographic forwarding, yet it has been commonly used in several routing protocol designs [15][16]. The prominence of Greedy is perhaps due to its use in the ground-breaking work of the GPSR[21] design, or due to the intuitive front it presents as a 'closest to the destination' next-hop selector. Nevertheless in comparison to Greedy, S* performance has proven to be a better next-hop selector that moves packets more quickly to the destination; thus offering better packets delivery rates.

Although the two algorithms' simulation results graphs display some congruent areas, separate differences also appeared in the performance patterns. The remarkably better performance of S* in the outgoing packets delivery rates makes it more recommendable for use during dispatch or on the sending-node side in VANET packets forwarding. Such an algorithm for the outgoing packets stage could read as below.

```
/*forward packets*/
if (source node or packets dispatching_period)
     {use S* forwarding}
else
     { …};
```

The observed differences in the results graphs of S* and Greedy sometimes show better activity of either of the algorithms in certain regions of the graphs; which also suggests that there is some advantage to be derived in the integrative or complementary use of both algorithms during a forwarding session. It may indeed be possible to switch between the techniques during a forwarding session in accordance with some specified conditions encountered. Again, an algorithm for a routine that constantly checks such suitability of conditions would be of the form shown below.

```
/*constantly check VANET current environment and assign technique*/
             if (VANET condition A)
                  {use S* forwarding}
             elseif (VANET condition B)
                  {use Greedy forwarding}
             elseif (…)
                  {use …}…
```

Hence, we propose that a mix of relevant greedy forwarding techniques, including others discussed in section 2.5 such as MFR, could be integrated as a *hybrid-greedy technique* that implement geographic location-aware forwarding tasks. Similar proposals have been made for the development of hybrid routing protocols [2][64][17][98].

In the current design of the S* technique, the forwarding program scans the destination node's present position for each packet at each hop. The high overhead resulting from such position scans could be reduced by tuning the period of acquisition to some optimized time intervals.

Then each periodic scan would serve several packets that pass through the concerned node within the determined interval.

5.2 Effectiveness of the orthogonal-to-baseline search space limitations

The results obtained concerning the *VDIST* vertical distance selection mechanism demonstrated the effectiveness of the tool, as seen largely in the graphs of the unicast incoming packets delivery rates. Among all the limits set for the unicast incoming Vdist graphs, as shown in Fig. 4.3 (a) - (e), the 500m Vdist most effectively delimits the multi-hop transmission range for the unicast packets forwarding in the 1/216000m^2 density network. It is surprising that the 500m restriction outcome is better than that of 900m; but this could be attributable to lesser collisions, leading to reduced loss of the unicast incoming packets [98] in the former case.

The unicast outgoing Vdist graphs of Fig. 4.4 (a) - (e) that exhibited uniformity could connote that the concerned source nodes were unaffected in their connectedness to their immediate next-hops which permits uninhibited packets dispatch. If so, this lack of inhibition is unexpected, especially in the case of the narrow 20m *VDIST_MAX* restriction that should have experienced much packets dropping due to probable instances of no qualified next-hop neighbour within range.

The *DIST* mechanism in general shows its utility in limiting the area of path search in VANET packets routing and forwarding schemes. The primary purpose for using such orthogonal-to-baseline search space limiting mechanism is to select only a subset of neighbour nodes from a most appropriate region within the transmission range of a forwarding node, from which an optimally efficient next-hop is then chosen. The *VDIST* mechanism can be adapted to even control neighbour list sizes so that fewer and only the most eligible next-hop nodes are involved.

We show some basic measures that extend the distance *DIST* mechanism when the method is to be applied in developing restricted neighbour lists. The radius distance mechanism, *RDIST*, which is depicted in Fig. 5.1, is a measure of the distance between a forwarding node and any neighbour node within its range. For any node *B* with position coordinates (x_B, y_B) that lies within the range of a reference node *A* with position coordinates (x_A, y_A), then the *RDIST* of *B* with respect to *A* is:

$$RDIST_{AB} = \sqrt{(x_B - x_A)^2 + (y_B - y_A)^2} \qquad (6.1)$$

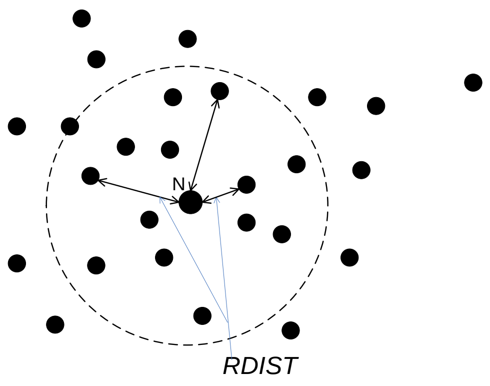


Figure 5.1 : *RDIST* radial distance measures of nodes within the range of node N

Then for any specified minimum radial limit *RDIST_MIN* or maximum limit *RDIST_MAX*, a qualifying neighbour's radial distance, *nei_rdist*, satisfies the condition:

$$RDIST\_MIN \leq nei\_rdist \leq RDIST\_MAX \qquad (6.2)$$

Similarly, with reference to the *VDIST_MAX* mechanism that we have hitherto used, we can have a minimum vertical distance to the baseline also as *VDIST_MIN*; then a qualifying neighbour's vertical distance, *nei_vdist*, satisfies the condition

$$VDIST\_MIN \leq nei\_vdist \leq VDIST\_MAX \qquad (6.3)$$

The authors in [80][101] have shown a requirement for adjustable transmission ranges in VANET, through the employment of requisite transmitter devices. Excessively overlapping transmission areas as shown in Fig. 5.2 is undesirable in VANET, since such a situation exacerbates network congestion and packets collisions. Nodes, together with their associated protocol, can therefore be designed to auto-sense density levels in VANET and adjust packets forwarding ranges accordingly; as such, the *VDIST* mechanism can serve as an alternative to the use of adjustable-range transmitter hardware. Thus, the transmission distances or effective transmission limits of nodes in VANET can be flexibly controlled and optimized for reduced packets collision effects in the network. This derivable *DIST* mechanism might be quite suitable in the NFP [31] greedy technique (in section 2.5.2) that must function with an adjustable radio transmission system would be useful.

Figure 5.2 (a) & (b): VANET scenario showing nodes transmission ranges

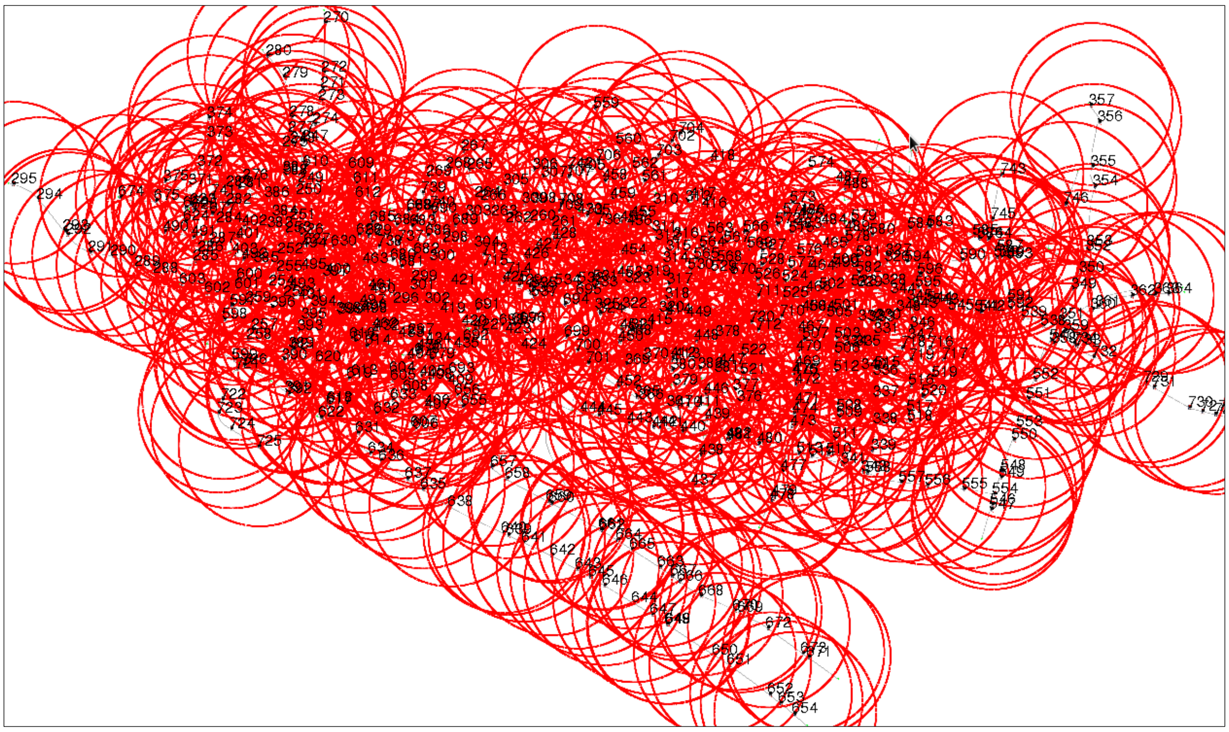


Fig. 5.2 (a) Highly overlapping transmission ranges of 885m radius

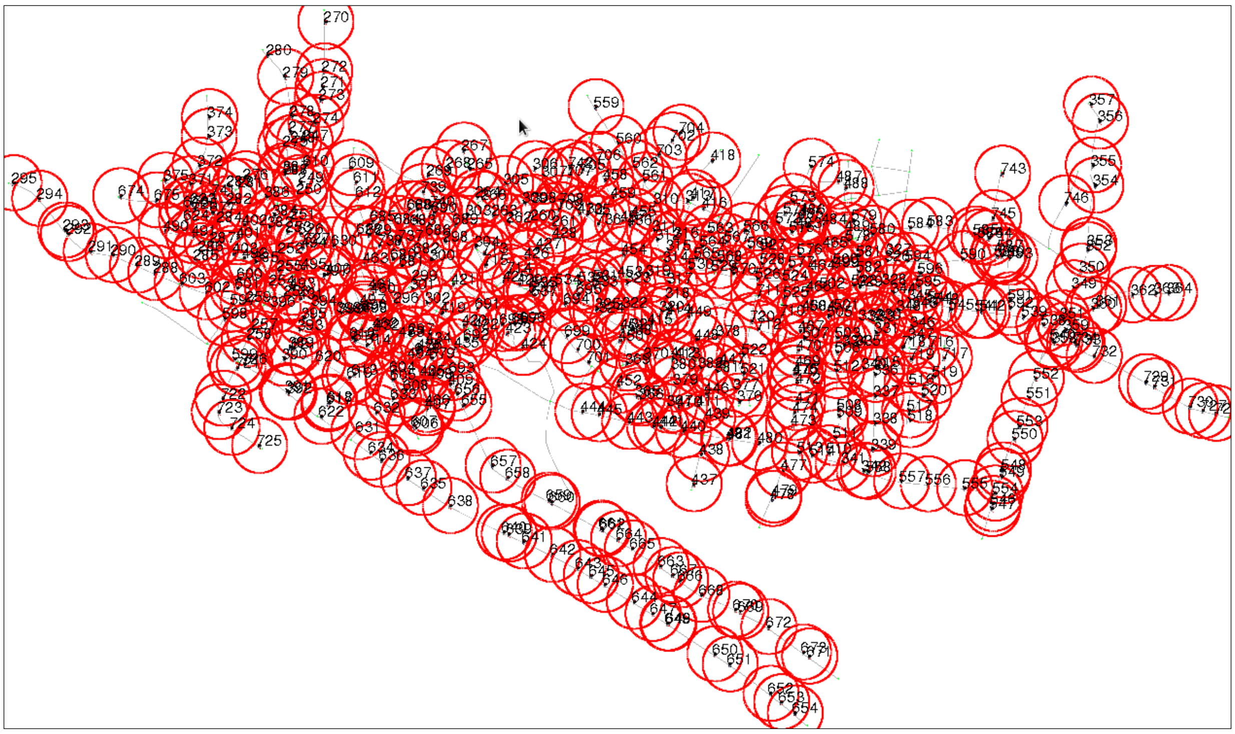


Fig. 5.2 (b) Moderately overlapping transmission ranges of 300m radius

As shown in Fig. 5.3, the *VDIST* mechanism is usable in fixing the *transmission distance* (or *forwarding range*) in a flexible manner when it is used to delimit the *transmission range* of a forwarding node.

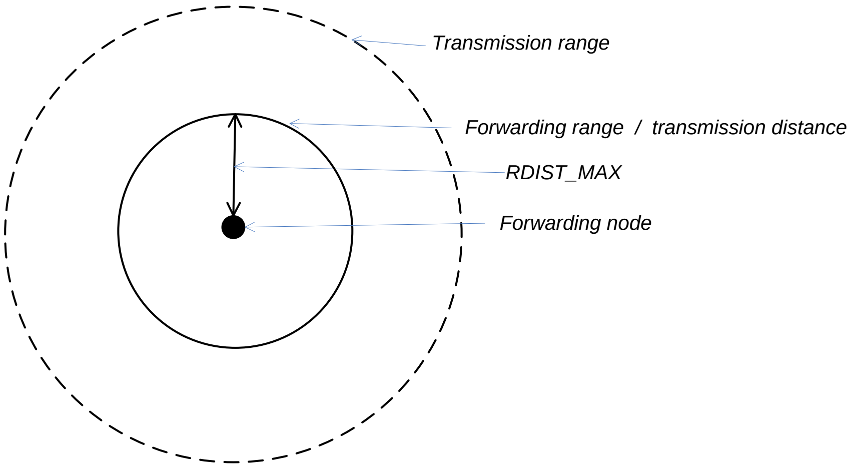


Figure 5.3: Transmission range delimiting using RDIST_MAX mechanism

In the figure, the inner circle denotes the currently fixed *effective transmission range* for the node. Again, the effective transmission range is determinable based on the prevailing VANET conditions, to offer maximal forwarding performance.

5.3 Case for dynamic hop-after-hop position-aware forwarding

The observed forwarding performance of the table-driven method in section 4.3 indicates the problem with this approach in VANET, which adversely affects packets delivery success rate. In table-driven systems, when a source has acquired the routing information to a destination, all packets stream serially along the pre-determined route of a multi-hop sequence of nodes. The same practice of following the path of a pre-set sequence of nodes also applies in routing methods where the route map is embedded in packet headers. This mappings approach is typically rigid and does not easily adjust to network reconfigurations to make better routing choices; even though the protocol may at some stage re-initiate route discovery as in the case of AODV [11]. The insensitivity or slow response of the method to positional changes results in distorted routing pathways, and packets dropping when the nodes in the routing sequence go out of the range of one another. Furthermore, much midway re-initiating of route discovery plans in the VANET environment incurs excessive overhead and severely affects throughput. Hence, the header-mapping approach that is characteristic of some location-aware MANET protocol designs such as LAR [23] and GLAR [19] categorically renders them unsuitable for VANET. Similarly, those protocols that are classified as greedy location-aware routing protocols for VANET, e.g. GSR and A-STAR [15][14], but which use header-mapping approach are unsuitable.

In contrast to the table-driven or header-mapping approaches, there is the class of location-aware greedy routing designs, such as is in GPSR [21] that embed the destination node positional and identity information in packets' headers for routing. The positional information is used at each forwarding node to determine the appropriate next-hop; in which case the decision-making is dynamic and flexible as it is performed for each packet at each forwarding node; allowing the choice of the next-hop to be made continually and in concert with the changing network configuration. The greedy method results, shown in section 4.3, reveals the value in VANET of this highly flexible routing method that selects next-hop on the fly. In comparison to the table-driven method, the demonstration (Fig. 4.6 (a) – (b)) of the greedy *packet swinging* approach (discussed in section 2.3.7) exhibits sustained packets streaming by ever adjusting itself to new sequencing of hops along source to destination path. The very dynamic path reconstructing exhibits an apparently unbroken connectivity leading to better packets delivery rates; compared to the table driven system that demonstrated long periods of connectivity severances leading to much packets dropping. We needed not show the graphs of packets delivery rates regarding these two methods as it has generally been proved [22] [42] that the table-driven systems perform poorly in VANET. However, we have shown the superior performance of packet swinging routing, rather than header mapping, within the geographic greedy forwarding practice.

5.4 Future work

The S* method exhibited distinctive packets forwarding performance in the VANET environment as has been shown by the results of the simulations. The efficacy of the S* technique in VANET needs to be further investigated to validate its applicability. What factors actually made S* to demonstrate significantly better results than Greedy forwarding in the study? Although the O(n) of S* is supposed to be slightly higher than that of Greedy since the former only performs an additional line of calculations in their almost similar codes; while based on this fact Greedy should have performed better or there should have been generally closely competing results rather than what we got. The understanding and more accurate explanations of the contributing factors to better S* performance can give us lead into tuning such phenomenon for improved performances still.

There are certain time windows during the routing span when Greedy performed extremely low while S* enormously well, and vice-versa; particularly in the turnout of the 15-node experiments. We attribute the very contrasting performances to situations of network configurations that suit either of the forwarding techniques best. We hope to understand and explain these situations in relation to each algorithm. The prospect of discovering and defining which configurations in VANET simulation suits which technique most might be a daunting task to dissolve due to the internal complexities of the environment. However, adequate perceptions of the inner workings of the varying environment are a requirement for designing hybrid-greedy systems.

The *VDIST* mechanism is best employable in dense networks, i.e. to curtail processing costs in congested environments. We need to examine the mechanism's performance in networks higher than the 1/216000m^2 density that we were limited. Although we hardcoded the parameters of the distances we used in testing the mechanism, it normally should select its limits automatically based on prevailing traffic volume. Another issue in the *VDIST* scheme that demands an explanation is the results (in Fig. 4.4 (a) - (e)) that showed no apparent effect in the unicast outgoing packets for the different set values of distance limits.

Although S* is a reactive technique, but in similarity to some other greedy approaches such as GPSR, the neighbour lists at the nodes are maintained by a proactive beaconing scheme. It will be interesting also to investigate the S* technique in a reactive beaconing environment as suggested by [21] in the case of GPSR. Reactive beaconing performs neighbours' position discovery just when a node has data packet(s) to forward, thus saving substantially on neighbour lists maintenance.

6.0 CONCLUSION

We have demonstrated the performance of the S* as an efficient geographic forwarding technique for routing in VANET. We compared the S* routing technique with the Greedy forwarding technique that have hitherto been employed in several VANET protocol designs. We designed S*, based on the popular A* path search method; and supplemented it with the *DIST* mechanism for limiting search-space for a forwarding node's potential next-hop neighbours. We utilized the EstiNet simulator to model and evaluate the S* method routing efficiency in packets delivery rates of unicast incoming and outgoing packets over varied network densities. The results surprisingly show the S* performance surpassing the Greedy method in packets delivery success rates.

VANET environments do vary as a vehicular node transits from one district to another and at different periods. These spatial and temporal variations in VANET demand that the applicable routing protocol should be flexible in adjusting to the changing conditions. Hence, as deduced from the results, we recommend a *hybridized-greedy* routing method that may include any suitable subset of the forwarding techniques of MFR, NFP, NC, Greedy, CR and S* for accomplishing optimum packets delivery in VANET.

The S* technique as a variant of the A* algorithm is a shortest path routing method that will generally provide some advantages over the Greedy algorithm when applied in VANET or MANET environments. An S* reduced-distance travel means quicker and higher packets delivery rate. Shorter path also lead to faster packets transfers that aids network decongestion as well as reducing transmission power consumptions [98]. A special feature of the S* algorithm path search method is that it evaluates only the immediate children (that are denoted as neighbours) of the forwarding node at each stage; and instantly discards the memory as soon as the packet is forwarded to the next hop. The marginal memory requirement advantage of S* is in contrast to the classic A* that requires large memory storage for maintaining its OPEN and CLOSED queues, which easily surges in large networks.

The adjoining *DIST* mechanism in the S* method demonstrated the effectiveness of its utility in VANET, especially in dense networks, such as are common in megacities, e.g. Lagos that is shown in Appendix F. A dense network implies a high level of neighbour lists' content at the nodes, with the associated high cost of maintenance especially in the topologically unstable VANET. Hence, the mechanism can support selection of neighbours from some defined region within the transmission range. We also show the potentials of the *DIST* mechanism as a tool for auto-adjusting and fixing maximum forwarding distance. The mechanism may therefore be a suitable substitute for the adjustable-range transmitter hardware that is required alongside the NFP next-hop selection technique. Similarly, it can serve for optimization of transmission range overlap in dense network, to minimize packets collisions.

Through our demonstration of the gross inefficiency of the table-driven systems, we reinforce the utility of the geographic location-aware method with emphasis on dynamic greedy forwarding, i.e. *packet swinging*. Routing over pre-computed paths is problematic in VANET due to its continuously changing topology. Whereas some study results, such as in the evaluation of GSR and A-STAR protocols, have revealed the problems of pre-computed routing path in VANET; yet the apparently better performance of these protocols against those that they were compared have not helped in properly isolating the path mapping problem in VANET. In the VANET environment, path-map in packet headers or table-described path quickly become obsolete and invalid as nodes rapidly change positions. Therefore methods that forward packets dynamically, which S* implement by design, are best suited for routing in VANET.

REFERENCES

[1] M. Conti and S. Giordano, “Mobile ad hoc networking: milestones, challenges, and new research directions,” *Communications Magazine, IEEE*, vol. 52, pp. 85–96, Jan. 2014.

[2] W. Chen, R. K. Guha, T. J. Kwon, J. Lee, and Y. Hsu, “A survey and challenges in routing and data dissemination in vehicular ad hoc networks,” *Wireless Communications and Mobile Computing*, vol. 11, no. 7, pp. 787–795, 2011.

[3] A. Skordylis and N. Trigoni, “Efficient data propagation in traffic-monitoring vehicular networks,” *Intelligent Transportation Systems, IEEE Transactions on*, vol. 12, no. 3, pp. 680–694, 2011.

[4] Sherali Zeadally, Ray Hunt, Yuh-Shyan Chen, Angela Irwin, and Aamir Hassan, “Vehicular ad hoc networks (VANETS): status, results, and challenges,” *Telecommunication Systems, Springer US*, vol. 50, no. 4, pp. 217–241, Aug. 2012.

[5] W Kellerer, C Bettstetter, C Schwingenschlogl, and P Sties, “(Auto) mobile communication in a heterogeneous and converged world,” *Personal Communications, IEEE*, vol. 8, no. 6, pp. 41–47, Dec-2001.

[6] E. Schoch, F. Kargl, M. Weber, and T. Leinmuller, “Communication patterns in VANETs,” *Communications Magazine, IEEE*, vol. 46, no. 11, pp. 119–125, 2008.

[7] Ciprian Dobre and George Cristian Tudor, “Mobile Advertisement in Vehicular Ad-Hoc Networks,” in *Proc. of 16th Annual Conference on Web Technology, New Media, Communications and Telematics Theory, Methods, Tools and Applications (Euromedia’2011)*, London, 2011, pp. 43–49.

[8] Katrin Sjöberg, “Vehicular ad hoc networks – challenges, opportunities and standardization,” Centre for Research on Embedded Systems, VOLVO, HALMSTAD, Apr-2012.

[9] Yousef-Awwad Daraghmi and Ivan Stojmenovic, “Forwarding methods in data dissemination and routing protocols for vehicular Ad Hoc networks,” *Network, IEEE*, vol. 27, no. 6, pp. 74–79, Dec. 2013.

[10] Reinaldo Bezerra Braga and Hervé Martin, “Understanding Geographic Routing in Vehicular Ad Hoc Networks,” in *GEOProcessing 2011*, Guadeloupe, France.

[11] C. E. Perkins and E. M. Royer, “Ad-hoc On-demand distance vector routing,” presented at the Mobile Computing Systems and Applications, 1999. Proceedings. WMCSA’99. Second IEEE Workshop on, 1999, pp. 90–100.

[12] David B. Johnson, David A. Maltz, Tomasz Imielinski, and Hank Korth, "Dynamic source routing in ad hoc wireless networks," in *Mobile Computing*, vol. 353, Springer US: Kluwer Academic Publishers, 1996, pp. 153–181.

[13] Yun-Wei Lin, Yuh-Shyan Chen, and Sing-Ling Lee, "Routing Protocols in Vehicular Ad Hoc Networks: A Survey and Future Perspectives," *Journal of Information Science and Engineering*, vol. 26, no. 3, pp. 913–932, May 2010.

[14] Kevin C. Lee, Uichin Lee, and Mario Gerla, "Survey of Routing Protocols in Vehicular Ad Hoc Networks," in *Advances in Vehicular Ad-Hoc Networks: Developments and Challenges*, Mohamed Watfa, Ed. Hershey New York: IGI Global, 2010.

[15] S.-H. Cha, M.-W. Ryu, and K.-H. Cho, "A Survey of Greedy Routing Protocols for Vehicular Ad Hoc Networks," *Smart Computing Review*, vol. 2, no. 2, pp. 125–137, 2012.

[16] J. Bernsen and D. Manivannan, "Greedy routing protocols for vehicular ad hoc networks," presented at the Wireless Communications and Mobile Computing Conference, 2008. IWCMC'08. International, 2008, pp. 632–637.

[17] M. Al-Rabayah and R. Malaney, "A New Scalable Hybrid Routing Protocol for VANETs," *IEEE Transactions on Vehicular Technology*, vol. 61, no. 6, pp. 2625–2635, 2012.

[18] V. Naumov and T. R. Gross, "Connectivity-aware routing (CAR) in vehicular ad-hoc networks," presented at the INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE, 2007, pp. 1919–1927.

[19] N.-C. Wang, J.-S. Chen, Y.-F. Huang, S.-M. Wang, and S. Chen, "A Greedy Location-Aided Routing Protocol for Mobile Ad Hoc Networks," presented at the WSEAS International Conference. Proceedings. Mathematics and Computers in Science and Engineering, 2009.

[20] P. K. Sahu, E.-K. Wu, J. Sahoo, and M. Gerla, "BAHG: Back-Bone-Assisted Hop Greedy Routing for VANET's City Environments," 2013.

[21] B. Karp and H.-T. Kung, "GPSR: Greedy Perimeter Stateless Routing for Wireless Networks," presented at the Proceedings of the 6th annual international conference on Mobile computing and networking, 2000, pp. 243–254.

[22] Mauve M, Widmer J, and Hartenstein H, "A survey on position-based routing in mobile ad hoc networks," *IEEE Network*, vol. 15, no. 6, pp. 30–39, Dec. 2001.

[23] Y.-B. Ko and N. H. Vaidya, "Location-aided routing (LAR) in mobile ad hoc networks," presented at the Proceedings of the 4th annual ACM/IEEE international conference on Mobile computing and networking, 1998, pp. 66–75.

[24] Christian Lochert, Hannes Hartenstein, Jing Tian, Holger Füßler, Dagmar Hermann, and Martin Mauve, "A Routing Strategy for Vehicular Ad Hoc Networks in City Environments," in *Proceedings. Intelligent Vehicles Symposium, 2003.*, Columbus OH USA, 2003, pp. 156–161.

[25] Boon-Chong Seet, Genping Liu, Bu-Sung Lee, Chuan-Heng Foh, Kai-Juan Wong, and Keok-Kee Lee, "A-STAR: A Mobile Ad Hoc Routing Strategy for Metropolis Vehicular Communications," in *Proceedings, Third International IFIP-TC6 Networking Conference*, Athens, Greece, 2004, vol. 3042, pp. 989–999.

[26] Moez Jerbi, Sidi-Mohammed Senouci, and Yacine Ghamri-Doudane, "Towards Efficient Routing in Vehicular Ad Hoc Networks," *UbiCC Journal - Special issue of UbiRoads 2007*, vol. Ubiroads07.

[27] Jing Zhao and Guohong Cao, "VADD: Vehicle-Assisted Data Delivery in Vehicular Ad Hoc Networks," in *25th IEEE International Conference on Computer Communications*, Barcelona, Spain, 2006, pp. 1–12.

[28] J. Zhao and G. Cao, "VADD: Vehicle-assisted data delivery in vehicular ad hoc networks," presented at the IEEE infocom, 2006, vol. 6, p. 57.

[29] Carlos de Morais Cordeiro and Dharma Prakash Agrawal, "Routing in Ad-Hoc Networks," in *Ad Hoc and Sensor Networks Theory and Applications*, 2nd ed., World Scientific, 2011, p. 664.

[30] Stefan Rührup, "Theory and Practice of Geographic Routing," in *Ad Hoc and Sensor Wireless Networks: Architectures, Algorithms and Protocols*, vol. 1, Hai Liu, Xiaowen Chu, and Yiu-Wing Leung, Eds. Bentham Science, 2009.

[31] H. Liu, A. Nayak, and I. Stojmenovic, *Geographic Routing in Wireless Sensor and Actuator Networks. In book: Wireless Sensor and Actuator Networks: Algorithms and Protocols for Scalable Coordination and Data Communication.* Wiley, 2010.

[32] A. Autere, "Extensions and Applications of the A* Algorithm," Dissertation for the degree of Doctor of Science in Technology, Helsinki University of Technology (TKK), Finland, 2005.

[33] L. H. O. Rios and L. Chaimowicz, "A survey and classification of a* based best-first heuristic search algorithms," in *Advances in Artificial Intelligence–SBIA 2010*, Brazil, 2010, pp. 253–262.

[34] Rafia Inam, Daniel Cederman, and Philippas Tsigas, "A* Algorithm for Graphics Processors," presented at the THIRD SWEDISH WORKSHOP ON MULTI-CORE COMPUTING - MCC10, Sweden, 2010.

[35] Tatsuya Ohshima,, Pipaporn Eumthurapojn,, Liang Zhao,, and Hiroshi Nagamochi, “An A∗ Algorithm Framework for the point-to-point Time-Dependent Shortest Path Problem,” in *9th International Conference, Revised Selected Papers*, Dalian, China, 2010, vol. 7033, pp. 154–163.

[36] N. Loulloudes, G. Pallis, and M. D. Dikaiakos, “The dynamics of vehicular networks in urban environments,” *arXiv preprint arXiv:1007.4106*, 2010.

[37] C. Sommer, I. Dietrich, and F. Dressler, “Realistic simulation of network protocols in vanet scenarios,” presented at the 2007 Mobile Networking for Vehicular Environments, 2007, pp. 139–143.

[38] F. J. Martinez, C. K. Toh, J.-C. Cano, C. T. Calafate, and P. Manzoni, “A survey and comparative study of simulators for vehicular ad hoc networks (VANETs),” *Wireless Communications and Mobile Computing*, vol. 99, no. 7, pp. 1189–1212, 2009.

[39] H. Boeglen, B. Hilt, P. Lorenz, J. Ledy, A.-M. Poussard, and R. Vauzelle, “A survey of V2V channel modeling for VANET simulations,” presented at the Conference on Wireless On-Demand Network Systems and Services (WONS), 2011 Eighth International, 2011, pp. 117–123.

[40] M. Fiore, J. Harri, F. Filali, and C. Bonnet, “Vehicular mobility simulation for VANETs,” presented at the Simulation Symposium, 2007. ANSS’07. 40th Annual, 2007, pp. 301–309.

[41] Elizabeth M. Belding-Royer, “Routing Approaches In Mobile Ad Hoc Networks,” in *Mobile Ad Hoc Networking*, Stefano Basagni, Marco Conti, Silvia Giordano, and Ivan Stojmenovic, Eds. New Jersey: John Wiley & Sons, Inc., IEEE Press, pp. 275–300.

[42] Maria Kihl, Mihail Sichitiu, Ted Ekeroth, and Michael Rozenberg, “Reliable Geographical Multicast Routing in Vehicular Ad-hoc Networks.” .

[43] Imrich Chlamtaca, Marco Conti, and Jennifer J.-N. Liu, “Mobile ad hoc networking: imperatives and challenges,” vol. 1, no. 1, pp. 13–64, Jul. 2003.

[44] Xiaobo Chen and Danya Yao, “An empirically comparative analysis of 802.11n and 802.11p performances in CVIS,” in *ITS Telecommunications (ITST)*, Taipei, 2012, pp. 848 – 851.

[45] Shimin Sun, Jonghyun Kim, Yunho Jung, and Keecheon Kim, “Zone-based Greedy Perimeter Stateless Routing for VANET,” presented at the International Conference on Information Networking, 2009. ICOIN 2009., Chiang Mai, 2009, pp. 1–3.

[46] J N Al-Karaki and A E Kamal, “Routing techniques in wireless sensor networks: a survey,” *Wireless Communications, IEEE*, vol. 11, no. 6, pp. 6–28, Dec-2004.

[47] Bob Gray, “Soldiers, Agents and Wireless Networks: A Report on a Military Application,” DoD, Manchester UK, Proceedings of the International Conference on Practical Application of Intelligent Agents and Multi-Agents, 5th (PAAM 2000), Apr. 2000.

[48] Gayathri Chandrasekaran, “VANETs: The Networking Platform for Future Vehicular Applications.” Department of Computer Science, Rutgers University, 2008.

[49] TOYOTA, “The Future Vehicular Network Architecture,” Toyota ITC, USA, 2010.

[50] Jang-Ping Sheu, “Routing Protocols, chapter 4,” National Tsing Hua University, Taiwan, Apr-2014.

[51] Kemal Akkaya and Mohamed Younis, “A Survey on Routing Protocols for Wireless Sensor Networks.” .

[52] Mark A. Perillo and Wendi B. Heinzelman, “Wireless Sensor Network Protocols.” .

[53] Wendi B. Heinzelman, Anantha P. Chandrakasan, and Hari Balakrishnan, “An Application-Specific Protocol Architecture for Wireless Microsensor Networks,” *IEEE Transactions on Wireless Communications*, vol. 1, no. 4, pp. 660 – 670, Oct. 2002.

[54] Fan Li and Yu Wang, “Routing in vehicular ad hoc networks: A survey,” *Vehicular Technology Magazine, IEEE*, vol. 2, no. 2, pp. 12–22, Jun. 2007.

[55] S. Basagni, I. Chlamtac, V. R. Syrotiuk, and B. A. Woodward, “A distance routing effect algorithm for mobility (DREAM),” presented at the Proceedings of the 4th annual ACM/IEEE international conference on Mobile computing and networking, 1998, pp. 76–84.

[56] Chiara Buratti, Andrea Conti, Davide Dardari, and Roberto Verdone, “An Overview on Wireless Sensor Networks Technology and Evolution,” *MDPI Sensors*, vol. 9, no. 9, 2009.

[57] Jennifer J.-N. Liu and Conti M, “Mobile Ad Hoc Networking with a View of 4G Wireless: Imperatives and Challenges,” in *Mobile Ad Hoc Networking*, Basagni S, Conti M, Giordano S, and Stojmenovic I, Eds. New Jersey: Wiley-IEEE Press, 2004, pp. 1–45.

[58] Charles E. Perkins and Pravin Bhagwat, “DSDV - Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers,” in *Proceedings of the conference on Communications architectures, protocols and applications*, London, 1994, pp. 234–244.

[59] Jacquet P, Muhlethaler P, Clausen T, Laouiti A, Qayyum A, and Viennot, L., “Optimized link state routing protocol for ad hoc networks,” in *IEEE International Multi Topic Conference, 2001*, Lahore, Pakistan, 2001, pp. 62 – 68.

[60] Jérôme Haerri, Fethi Filali, and Christian Bonnet, “Performance Comparison of AODV and OLSR in VANETs Urban Environments under Realistic Mobility Patterns,” in *Med-Hoc-Net 2006,* Lipari, Italy, 2006.

[61] Spaho E, Ikeda M, Barolli L, Xhafa F, Younas M, and Takizawa M, “Performance of OLSR and DSDV Protocols in a VANET Scenario: Evaluation Using CAVENET and NS3,” in *2012 Seventh International Conference on Broadband, Wireless Computing, Communication and Applications (BWCCA),*, Victoria BC Canada, 2012, pp. 108 – 113.

[62] M Bouhorma, H Bentaouit, and A Boudhir, “Performance Comparison of Ad-hoc Routing Protocols AODV and DSR,” in *International Conference on Multimedia Computing and Systems, 2009,* Ouarzazate, Morocco, 2009, pp. 511–514.

[63] A Tuteja, R Gujral, and S thalia, “Comparative Performance Analysis of DSDV, AODV and DSR Routing Protocols in MANET Using NS2,” presented at the 2010 International Conference on Advances in Computer Engineering (ACE), Bangalore, Karnataka, India, 2010, pp. 330–333.

[64] Z J Haas, “A New Routing Protocol for the Reconfigurable Wireless Networks,” presented at the 1997 IEEE 6th International Conference on Universal Personal Communications Recordx, San Diego CA, 1997, vol. 2, pp. 562–566.

[65] Marwa Altayeb and Imad Mahgoub, “A Survey of Vehicular Ad hoc Networks Routing Protocols,” *International Journal of Innovation and Applied Studies,* vol. 3, no. 3, pp. 829–846, Jul. 2013.

[66] Holger Füßler, Martin Mauve, Hannes Hartenstein, and Michael Käsemann, “Location-Based Routing for Vehicular Ad-Hoc Networks,” presented at the MobiHoc ’02, Atlanta Georgia, 2002.

[67] Christian Lochert, Martin Mauve, Holger Füßler, and Hannes Hartenstein, “Geographic Routing in City Scenarios,” *ACM SIGMOBILE Mobile Computing and Communications Review,* vol. 9, no. 1, pp. 69–72, Jan. 2005.

[68] Valery Naumov, Rainer Baumann, and Thomas Gross, “An Evaluation of InterVehicle Ad Hoc Networks Based on Realistic Vehicular Traces,” in *7th ACM international symposium on Mobile ad hoc networking and computing Pages 108-119,* New York USA, 2006.

[69] Gregory G. Finn and Joseph D. Touch, “Network Construction and Routing in Geographic Overlays.” USC/Information Sciences Institute, Jul-2002.

[70] Qixiang Sun and Hector Garcia-Molina, “Using Ad-hoc Inter-vehicle Networks For Regional Alerts,” Stanford, 2005.

[71] Sushant Jain, Kevin Fall, and Rabin Patra, “Routing in a Delay Tolerant Network,” in *Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications,* Portland OR USA, 2004, pp. 145–158.

[72] Yuh-Shyan Chen and Yun-Wei Lin, “Routing Protocols in Vehicular Ad Hoc Networks,” in *Telematics Communication Technologies and Vehicular Networks: Wireless Architectures and Applications*, IGI Global, 2010, pp. 206–228.

[73] Hao Wu, Richard Fujimoto, Randall Guensler, and Michael Hunter, “MDDV: A Mobility-Centric Data Dissemination Algorithm for Vehicular Networks,” in *Proceedings of the 1st ACM international workshop on Vehicular ad hoc networks,* Philadelphia, PA, USA, 2004.

[74] “Greedy algorithm,” *Wikipedia*. Wikimedia Foundation, Inc., Jan-2014.

[75] S S Manvi, M S Kakkasageri, and C V Mahapurush, “Performance Analysis of AODV, DSR, and Swarm Intelligence Routing Protocols In Vehicular Ad hoc Network Environment,” presented at the 2009 International Conference on Future Computer and Communication, Kuala Lumpar, 2009, pp. 21–25.

[76] J. Fukuyama, “A probabilistic protocol for multihop routing in VANETs,” *Journal of Electrical and Computer Engineering,* vol. 2010, p. 42, 2010.

[77] Jing Tian and Lu Han, “Spatially aware packet routing for mobile ad hoc inter-vehicle radio networks,” in *Intelligent Transportation Systems, 2003.*, 2003, vol. 2, pp. 1546 – 1551.

[78] Athanasiso Vasilakos, Yan Zhang, and Thrasyvoulos V. Spyropoulos, Eds., “Message Dissemination in Vehicular Networks,” in *Delay Tolerant Networks: Protocols and Applications,* Boca Raton FL USA: Auerbach Publications, CRC Press, 2011.

[79] Katsaros K, Dianati M, and Long Le, “Effective implementation of location services for VANETs in hybrid network infrastructures,” in *2013 IEEE International Conference on Communications Workshops (ICC),*, Budapest, 2013, pp. 521 – 525.

[80] Ting-Chao Hou and Victor O K Li, “Transmission Range Control in Multihop Packet Radio Networks,” *IEEE Transactions on Communications*, vol. COM-34, no. 1, pp. 38–44, 1986.

[81] Ivan Stojmenovic and Xu Lin, “Power aware localized routing in wireless networks,” *IEEE Transactions on Parallel and Distributed Systems,* vol. 12, no. 11, pp. 1122–1133, Nov. 2001.

[82] Evangelos Kranakis, Harvinder Singh, and Jorge Urrutia, “Compass Routing on Geometric Networks,” in *11 TH CANADIAN CONFERENCE ON COMPUTATIONAL GEOMETRY,* Vancouver, 1999, pp. 51–54.

[83] D. Szer, F. Charpillet, and S. Zilberstein, "MAA*: A heuristic search algorithm for solving decentralized POMDPs," *arXiv preprint arXiv:1207.1359*, 2012.

[84] Ismail Chabini and Shan Lan, "Adaptations of the A* algorithm for the computation of fastest paths in deterministic discrete-time dynamic networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 3, no. 1, pp. 60–71, Mar. 2002.

[85] Razvan Stanica, Emmanuel Chaput, and André-Luc Beylot, "Simulation of vehicular ad-hoc networks: Challenges, review of tools and recommendations," *Computer Networks, Elsevier, Science Direct*, vol. 55, no. 14, pp. 3179–3188, Oct. 2011.

[86] M. Piórkowski, M. Raya, A. Lezama Lugo, P. Papadimitratos, M. Grossglauser, and J.-P. Hubaux, "TraNS: realistic joint traffic and network simulator for VANETs," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 12, no. 1, pp. 31–33, Jan-2008.

[87] Sommer C and Dressler F, "Progressing toward realistic mobility models in VANET simulations," *IEEE Communications Magazine*, vol. 46, no. 11, pp. 132–137, Nov. 2008.

[88] Yanmin Zhu, Chao Chen, and Min Gao, "An evaluation of vehicular networks with real vehicular GPS traces," *EURASIP Journal on Wireless Communications and Networking*, Jul. 2013.

[89] Evjola Spaho, Leonard Barolli, Gjergji Mino, Fatos Xhafa, and Vladi Kolici, "VANET Simulators - A survey on Mobility and Routing Protocols," in *2011 International Conference on Broadband and Wireless Computing, Communication and Applications (BWCCA)*, Bacelona, 2011, pp. 1–10.

[90] David R. Choffnes and Fabián E. Bustamante, "STRAW - An Integrated Mobility and Traffic Model for VANETs," in *Proceedings of the 2nd ACM international workshop on Vehicular ad hoc networks*, pp. 69 – 78.

[91] Li Zhiyuan and Hu Jinhong, "Framework of Real VANET Simulation Research," in *2011 Third International Conference on Multimedia Information Networking and Security (MINES)*, Shanghai, 2011, pp. 136–140.

[92] Marco Fiore and Jérôme Härri, "The Networking Shape of Vehicular Mobility," in *roceedings of the 9th ACM international symposium on Mobile ad hoc networking and computing*, Hong Kong, 2008, pp. 261–272.

[93] "ns (simulator)," *Wikipedia*. Wikimedia Foundation, Inc., Mar-2014.

[94] Christoph Sommer, Reinhard German, and Falko Dressler, "Bidirectionally Coupled Network and Road Traffic Simulation for Improved IVC Analysis," *IEEE TRANSACTIONS ON MOBILE COMPUTING*, vol. 10, no. 1, pp. 3–15, Jan. 2011.

[95] EstiNet, “EstiNet Sales Quotation Document.” EstiNet, 23-Aug-2013.

[96] Shie-Yuan Wang, Chih-Liang Chou, and Chun-Ming Yang, “EstiNet OpenFlow Network Simulator and Emulator,” *IEEE Communications Magazine*, vol. 51, no. 9, pp. 110–117, Sep. 2013.

[97] EstiNet, “Introduction of EstiNet Network Simulator.” Technologies Inc., 2013.

[98] Brad Nelson Karp, “Geographic Routing for Wireless Networks,” Doctor of Philosophy in the subject of Computer Science, Harvard University, Cambridge, Massachusetts, 2000.

[99] “A* search algorithm,” *Wikipedia*. Wikimedia Foundation, Inc.,, Mar-2014.

[100] Luis Felipe Urquiza Aguiar, “Design and implementation of routing protocols with anonymity for vehicular ad-hoc networks in urban environments,” Master’s Thesis, Universitat Politecnica De Catalunya, Barcelonatech, Barcelona, Spain, 2012.

[101] Hideaki Takagi and Leonard Kleinrock, “Optimal Transmission Ranges for Randomly Distributed Packet Radio Terminals,” *IEEE TRANSACTIONS ON COMMUNICATIONS,*, vol. COM -32, no. 3, pp. 246–257, Mar. 1984.

APPENDICES

Appendix A: S* Forwarding code sections

Appendix A1: Usermodule02.cc C/C++ file content showing some code sections

```
/*
 * Copyright (c) from 2000 to 2013
 *
 * EstiNet Technologies Inc.
 * All Rights Reserved.
 *
 * This source code file is part of the EstiNet network simulator and emulator.
 * It is an intellectual property of EstiNet Technologies Inc..
 * Without written permissions obtained from EstiNet Technologies Inc., it
 * is prohibited by law to disclose, transmit, post, or distribute this
 * source code file to any one or any place on the Internet that is
 * unauthorized by EstiNet Technologies Inc. to receive this source code file.
 *
 * 3/31/2011
 */

/*
 *    Ad hoc On-Demand Distance Vector (UserModule02) Routing
 *    reference: draft-ietf-manet-UserModule02-12.txt
 */

#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <fcntl.h>
#include <sys/uio.h>
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#include <string>
#include <assert.h>
#include <regTable.h>
#include <scheduler.h>
#include <typTable.h>
#include <estinet_api.h>
#include <ethernet.h>
#include <ip.h>
#include <random.h>
#include <module/user-defined/user_module_02.h>
#include <modBinder.h>
#include <math.h> ////atanda


using namespace UserModule02d;

#define LINK_LAYER_RETRY
/* #define LINK_LAYER_DROP */

```

```

MODULE_GENERATOR(UserModule02);

UserModule02::UserModule02(u_int32_t type, u_int32_t id, struct plist* pl, const char *name)
            : NslObject(type, id, pl, name)
{
      s_flowctl = DISABLED;
      r_flowctl = DISABLED;

  rreq_id = 0; /* ? */
  link_fail_list.slh_first = 0;
      bcache.slh_first = 0;

  acc_rreq = 0;
  acc_rerr = 0;

  qcur_ =0;
  rd_head = rd_tail = 0;
  qmax_ = 5;

  /* bind input file name */
  vBind_int("HELLO_INTERVAL",      &HELLO_INTERVAL);
      vBind_int("ALLOWED_HELLO_LOSS",   &ALLOWED_HELLO_LOSS);
  vBind_int("ACTIVE_ROUTE_TIMEOUT", &ACTIVE_ROUTE_TIMEOUT);
  vBind_int("DELETE_PERIOD",       &DELETE_PERIOD);
  vBind_int("NET_DIAMETER",        &NET_DIAMETER);
  vBind_int("NODE_TRAVERSAL_TIME",  &NODE_TRAVERSAL_TIME);
  vBind_int("RREQ_RETRIES",        &RREQ_RETRIES);
  vBind_int("RREQ_RATELIMIT",      &RREQ_RATELIMIT);
  vBind_int("RERR_RATELIMIT",      &RERR_RATELIMIT);

  vBind_int("VDIST_MAX",          &VDIST_MAX); ///////////////////atanda////////////////
}

UserModule02::~UserModule02() {
}

int UserModule02::init() {

  mip = GET_REG_VAR(get_port(), "IP", u_long *);

  Rt_entry *r = new Rt_entry;

  // the first entry is for its own
  r->rt_dst     = *mip;
  r->rt_nexthop = *mip;
  r->rt_valid_dst_seqno = true;
  r->rt_seqno   = 1;
  r->rt_hopcount= 0;
  r->rt_flags   = RTF_VALID;
  r->rt_time    = INFINITY_LIFETIME;

  ////////////////////////atanda, initialize next-hop
  nexthop_AStar = *mip;
  my_NID = get_nid();
  //GetNodeLoc(my_NID, my_X, my_Y, my_Z);


  rtable.insert(r);

  MILLI_TO_TICK(hello_interval_,  (u_int64_t)HELLO_INTERVAL);
  MILLI_TO_TICK(active_route_timeout_, (u_int64_t)ACTIVE_ROUTE_TIMEOUT);
  MILLI_TO_TICK(node_traversal_time_, (u_int64_t)NODE_TRAVERSAL_TIME);
  MILLI_TO_TICK(delete_period_, (u_int64_t)DELETE_PERIOD);

  MY_ROUTE_TIMEOUT = 2 * ACTIVE_ROUTE_TIMEOUT;
  MILLI_TO_TICK(my_route_timeout_, (u_int64_t)MY_ROUTE_TIMEOUT);
  NET_TRAVERSAL_TIME = (3 * NODE_TRAVERSAL_TIME * NET_DIAMETER /2);
  MILLI_TO_TICK(net_traversal_time_, (u_int64_t)NET_TRAVERSAL_TIME);
  PATH_DISCOVERY_TIME = (2 * NET_TRAVERSAL_TIME);
  MILLI_TO_TICK(path_discovery_time_,  (u_int64_t)PATH_DISCOVERY_TIME);

  MILLI_TO_TICK(route_check_timer_,  (u_int64_t)ROUTE_CHECK);
  MILLI_TO_TICK(rreq_check_timer_,  (u_int64_t)PENDING_RREQ_CHECK);
  MILLI_TO_TICK(recent_rreq_list_timer_,  (u_int64_t)RECENT_RREQ_LIST_CHECK);
  MILLI_TO_TICK(accumulated_rreq_rerr_timer_,  (u_int64_t)ACCUMULATED_RREQ_RERR_TIMER);
  MILLI_TO_TICK(nei_list_check_timer_,  (u_int64_t)NEI_LIST_CHECK);
  MILLI_TO_TICK(sendhello_timer_,  (u_int64_t)SENDHELLO_TIMER);
  MILLI_TO_TICK(link_fail_list_check_timer_,  (u_int64_t)LINK_FAIL_LIST_CHECK);
  /* MILLI_TO_TICK(printLoc_timer_,  (u_int64_t)5000); */

  BASE_OBJTYPE(type);
  type = POINTER_TO_MEMBER(UserModule02, sendHello);
  SendHello_timer.setCallOutObj(this, type);
  SendHello_timer.start((sendhello_timer_ + Random()%100000), 0);

int UserModule02::RREQ_retry(){



int UserModule02::CheckRecentRREQ() {
int UserModule02::ClearAccRREQ_RERR() {
int UserModule02::CheckNeiList() {
int UserModule02::CheckLinkFailList() {
int UserModule02::recv(ePacket_ *pkt) {

  u_long dst_ip, src_ip;

  Packet *p;
  struct UserModule02_packet *my_pkt;
  Rt_entry *dst_route;
  /* int lastseqno; */

  assert(pkt&&(p=(Packet *)pkt->DataInfo_));
  GET_PKT(p, pkt);

  char pkttype[5];
  strncpy(pkttype, p->pkt_get(), 4);
  pkttype[4]='\0';

  my_pkt = (struct UserModule02_packet *)p->pkt_get();


  if (strcmp(pkttype,"AODV") == 0){
      dst_ip = my_pkt->dst_ip;
    src_ip = my_pkt->src_ip;
  }
  else {
    IP_DST(dst_ip, p->pkt_sget());
    IP_SRC(src_ip, p->pkt_sget());

  }
//////////////atanda, BEGIN-if it is not UserModule02 protocol packet
  /*  Receive normal packet, we must help it forward to
  *   next node ,or if it is my packet, I pass it to interface layer.
  */
  if (bcmp(my_pkt->pro_type, "AODV", 4) != 0 )
  {
      if ((dst_ip == *mip) || is_ipv4_broadcast(get_nid(), dst_ip))
    {
      return (put(pkt, recvtarget_));
    } else
     {
      ///////////atanda, BEGIN-ASSIGNMENT OF A* NEXTHOP
      nexthop_AStar = getnexthopAStar(dst_ip);
```

```

        p->rt_setgw(nexthop_AStar);
        p->pkt_setflow(PF_SEND);
        return (sendToQueue(pkt)); //atanda, END-ASSIGNMENT OF A* NEXTHOP

        /* hwchu:
         *   There is no chance for this packet to enter the kernel,
         *   so we decrement its TTL here.
         */
        u_char ttl;

        GET_IP_TTL(ttl, p->pkt_sget());
        if (ttl <= 1)
        {
          return put(pkt, recvtarget_);
        }
        IP_DEC_TTL(p->pkt_sget());
        }
/////////////////////////atanda, rt and rtable issues.
//atanda, BEGIN-CODE DISABLED
//    int lookup_result = rtable.rt_lookup(dst_ip);
//    }else if (lookup_result == RT_NOT_EXIST){
//    }
//    freePacket(pkt);
//    return (1);
// ///////////atanda, END-if it is not UserModule02 protocol packet
////////////////atanda, END-CODE DISABLED
   }
   if (bcmp(my_pkt->pro_type, "AODV_RREQ", 10) == 0){

        struct RREQ_msg  *my_rreq;

        // update the neighbor list
        nei_list.update(my_rrep->rrep_dst_addr, (GetCurrentTime() + (ALLOWED_HELLO_LOSS * hello_interval_)));
      // not RREQ,RREP,RERR packet
      else{
      printf("[%u]: receive AODV_unknown pkt (type:%s) at tick=%llu\n",
          get_nid(), my_pkt->pro_type, GetCurrentTime());
      //assert(0);
      return 1;
      }

      freePacket(pkt);
      return (1);

}


int
UserModule02::send(ePacket_ *pkt) {

     Packet     *p;
     u_long     dst_ip, src_ip;

     assert(pkt&&(p=(Packet *)pkt->DataInfo_));

     GET_PKT(p, pkt);

     IP_DST(dst_ip, p->pkt_sget());
     IP_SRC(src_ip, p->pkt_sget());

////////////////atanda, BEGIN-this code segment was copied from recv() to be able to use my_pkt to distinct non-UserModule02 with if-else condition down below///////
  struct UserModule02_packet *my_pkt;
  my_pkt = (struct UserModule02_packet *)p->pkt_get();

//////////////////////END-this code segment was copied from recv()////////////////
```

```

    if (is_ipv4_broadcast(get_nid(), dst_ip)){
      // It's a broadcast pkt. Just send it.
      sendToQueue(pkt);
      return 1;
    }

    int lookup_result = rtable.rt_lookup(dst_ip);
    if (lookup_result != RTF_VALID) {

          if (!ctrl_table.ifExist(dst_ip)) {

      ctrl_table.insert(dst_ip, GetCurrentTime()+net_traversal_time_);
      if (ctrl_table.attachPkt(dst_ip, pkt) < 0) {
        freePacket(pkt);
        return (1);
      }

      if (acc_rreq <= RREQ_RATELIMIT) {
              sendRREQ(dst_ip, NET_DIAMETER);
        acc_rreq++;
      }
      return (1);

          }else{
        if (ctrl_table.attachPkt(dst_ip, pkt) < 0)
          freePacket(pkt);
        return (1);
   }
    }
    else {

///////atanda, BEGIN-set in if-else//////////////////////////////////////////
   if (bcmp(my_pkt->pro_type, "AODV", 4) != 0 )
   {
      nexthop_AStar = getnexthopAStar(dst_ip);
      p->rt_setgw(nexthop_AStar);
      sendToQueue(pkt);
   }///////atanda, END-set in if///////////////////////////////
   else
   {
      //atanda, this is existing code segment
            Rt_entry *rt0 = rtable.rt_get(dst_ip);

      // each time the route is used for transmission,
      // update its lifetime
      rt0->rt_time = GetCurrentTime() + active_route_timeout_;

      p->rt_setgw(rt0->rt_nexthop);
          sendToQueue(pkt);
   }
   return (1);
int UserModule02::sendToQueue(ePacket_ *pkt) {

    int     (NslObject::*upcall)(MBinder *);
               /* Do flow control for myself with s_queue */
 *   Regulary Send Broadcast HELLO message.
 */
int UserModule02::sendHello() {

//atanda, though this function is not disabled, it is disabled where it is called; and so it is for some others
int UserModule02::updateRT(u_long dst, u_long nexthop, u_int32_t seqno, u_int16_t hopcount, u_int64_t lifetime) {

}

///////////////////////////////////////atanda, ASTAR ENGINE CODE///////////////////////////////////////
int UserModule02::getnexthopAStar(u_long dst_ip)
```

```
{
  //initialize to myself
  u_long bestnexthop = *mip;

  int dst_NID;
  double dst_X, dst_Y, dst_Z;


  dst_NID = ipv4addr_to_nodeid(dst_ip);

  GetNodeLoc(my_NID, my_X, my_Y, my_Z);
  GetNodeLoc(dst_NID, dst_X, dst_Y, dst_Z);

  //BEGIN Calculate baseline values
  double a, b, c;

  /* Compute the aX + bY + C = 0 line equation formed by (x0, y0) and (x1, y1). //atanda, template obtained from obstacles.cc*/
     if (dst_Y == my_Y)
  {
           a = 0;
           b = 1;
           c = -1 * my_Y;
           if (dst_X == my_X)
                 a = b = c = 0;
     }
     else if (dst_X == my_X)
  {
           a = 1;
           b = 0;
           c = -1 * my_X;
     } else
  {
           b = -1;
           a = (dst_Y - my_Y) / (dst_X - my_X);
           c = my_Y - my_X * a;
     } ///END Calculate baseline values

  //my estimate straight line distance to destination
  double my_h = sqrt(((my_X - dst_X)*(my_X - dst_X)) + ((my_Y - dst_Y)*(my_Y - dst_Y)));
  int nei_NID;
  double nei_f, nei_g, nei_h;   //For f = g + h
  double least_f = 10000000;    //initialize with an infinite big value
  double nei_vdist;
  double nei_X, nei_Y, nei_Z;

  Nei_entry *p_nei = nei_list.getHead();

  //BEGIN Traverse list of my neighbors and make selection
  while (p_nei)
  {
    nei_NID = ipv4addr_to_nodeid(p_nei->nei_addr);

    if (nei_NID == dst_NID)
    {
      bestnexthop = p_nei->nei_addr;
      break;
    }
    else //assess for nexthop
    {
    GetNodeLoc(nei_NID, nei_X, nei_Y, nei_Z);

    ///perpendicular distance to baseline, VDST = |((aX+bY+c)/sqrt((a*a)+(b*b)))|
    nei_vdist = fabs(((a * nei_X) + (b * nei_Y) + c) / sqrt((a * a) + (b * b)));

    nei_h = sqrt(((nei_X - dst_X)*(nei_X - dst_X)) + ((nei_Y - dst_Y)*(nei_Y - dst_Y)));

    //positive advance and restrict range of nodes allowed from baseline
```

```
      if ((nei_h < my_h) && (nei_vdist < VDIST_MAX))  // int VDIST_MAX .h file
      {
        //Calculate f = distance g + distance h
        nei_g = sqrt(((my_X - nei_X)*(my_X - nei_X)) + ((my_Y - nei_Y)*(my_Y - nei_Y)));
        nei_f = nei_g + nei_h;

        if (nei_f < least_f)
        {
          least_f = nei_f;
          bestnexthop = p_nei->nei_addr;
        }
      }
    }
    p_nei = p_nei->next;
  }
  return bestnexthop;
}//getnexthopAStar(



/////////////////////////////BEGIN- PASTED odv.rt.cc FILE FROM HERE////////////////////////////////////////////////
/// Nei_entry /////////////////////////////////
Nei_entry::Nei_entry(u_long addr, u_int64_t lifetime) {
  nei_addr = addr;
  nei_time = lifetime;
  next = NULL;
}

Nei_entry::~Nei_entry() {
}

/// Neighbors /////////////////////////////////
Neighbors::Neighbors() {
  nei_head = NULL;
}

Neighbors::~Neighbors() {
  Nei_entry *n = nei_head;
  Nei_entry *tmp;
bool CtrlTable::ifExist(u_long addr) {

u_char Unreach_list::unreach_count() {
  return unr_count;
}

//////////////////////////////END- PASTED odvrt.cc//////////////////////////////////////////////


```

Appendix A2: Usermodule02.h header file content showing some code sections

```
/*
 * Copyright (c) from 2000 to 2013
 *
 * EstiNet Technologies Inc.
 * All Rights Reserved.
 *
 * This source code file is part of the EstiNet network simulator and emulator.
 * It is an intellectual property of EstiNet Technologies Inc..
 * Without written permissions obtained from EstiNet Technologies Inc., it
 * is prohibited by law to disclose, transmit, post, or distribute this
 * source code file to any one or any place on the Internet that is
 * unauthorized by EstiNet Technologies Inc. to receive this source code file.
 *
 * 3/31/2011
 */

#ifndef __user_module_02_h__
#define __user_module_02_h__

#include <event.h>
#include <object.h>
#include <mylist.h>
#include <timer.h>
#include <packet.h>
//#include <route/UserModule02/mstate.h>

#define INFINITY_HOPCNT  0xff
#define INFINITY_LIFETIME  0x7fffffff
#define UserModule02_MAXQUEUELEN 30

#define TTL_THRESHOLD 7

#define LINK_FAIL_LIFETIME          500   // ms
#define LINK_FAIL_THRESHOLD          4    // times

// millisec
#define SENDHELLO_TIMER             50
#define ROUTE_CHECK                 300
#define PENDING_RREQ_CHECK          50
#define RECENT_RREQ_LIST_CHECK      300
#define ACCUMULATED_RREQ_RERR_TIMER  1000
#define NEI_LIST_CHECK              300
#define LINK_FAIL_LIST_CHECK        1000




namespace UserModule02d{

class UserModule02;

/*
 *   Neighbors entry and Neighbors List
 */

class Nei_entry {
public:
      u_long          nei_addr;     // ip address
      u_int64_t           nei_time;     // expire time

      Nei_entry           *next;
public:
      Nei_entry(u_long addr, u_int64_t lifetime);
```

```
    ~Nei_entry();
};

class Neighbors {
private:
  Nei_entry         *nei_head;

public:
  Neighbors();
  ~Neighbors();

class UserModule02 : public NslObject {
                              // default
      int  HELLO_INTERVAL;       //1000
      int  ALLOWED_HELLO_LOSS ;   //2
      int  ACTIVE_ROUTE_TIMEOUT;  //3000
      int  DELETE_PERIOD;        //3000
  int  NET_DIAMETER;         //15
  int  NODE_TRAVERSAL_TIME;   //40 ms
  int  RREQ_RETRIES;         //5
  int  RREQ_RATELIMIT;       //10/per sec
  int  RERR_RATELIMIT;       //10/per sec

  int  VDIST_MAX;         // 1000 default   /////////atanda/////////////////

      int  MY_ROUTE_TIMEOUT;
  int  NET_TRAVERSAL_TIME;
  int  PATH_DISCOVERY_TIME;

 private:
     timerObj    SendHello_timer;
  timerObj    DelHello_timer;
  timerObj              RT_timer;
  timerObj              SendRREQ_timer;
  timerObj              RecentRREQ_timer;
  timerObj              AccRREQ_RERR_timer;
  timerObj              Nei_List_timer;
  timerObj              PrintLoc_timer;

  // interval tmp variable
     u_int64_t             hello_interval_;
     u_int64_t             delete_period_;

  u_int64_t             active_route_timeout_;
  u_int64_t             my_route_timeout_;
  u_int64_t             node_traversal_time_;
  u_int64_t             net_traversal_time_;
  u_int64_t             path_discovery_time_;

  u_int64_t             route_check_timer_;
  u_int64_t             rreq_check_timer_;
  u_int64_t             recent_rreq_list_timer_;
  u_int64_t             accumulated_rreq_rerr_timer_;
  u_int64_t             nei_list_check_timer_;
  u_int64_t             sendhello_timer_;
  u_int64_t             link_fail_list_check_timer_;
  u_int64_t             printLoc_timer_;

     u_long        *mip;             // my IP address

     int                  rreq_id;           // my rreq ID

  //////////////////atanda////////////////////
  u_long        nexthop_AStar;
  int        my_NID;
  double        my_X, my_Y, my_Z;

```

```

   UserModule02_RtTable         rtable;           // my Routing Table
   CtrlTable            ctrl_table;        // my UserModule02queue table
 LocalRepairTable       local_repair_table;
   SLIST_HEAD( ,Link_fail_entry)  link_fail_list;
   SLIST_HEAD( ,BroadcastID)  bcache;         // Broadcase ID cache
 Neighbors             nei_list;         // neighbor list

 // accumulated count for RREQ/RERR, will be clean to 0 every second
 int                   acc_rreq;
 int                   acc_rerr;

   int                   qmax_;            // UserModule02queue's max
 int                   qcur_;            // UserModule02queue current count
 ePacket_              *rd_head;         // UserModule02queue's head
 ePacket_              *rd_tail;


 public:

   UserModule02(u_int32_t type, u_int32_t id, struct plist* pl, const char *name);
   ~UserModule02();

   int                   init();
   int                   recv(ePacket_ *pkt);
   int                   send(ePacket_ *pkt);
   int         sendHello();
 int         miew();

 int         HelloTimer();
 int                   RTTimer();
 int                   RREQ_retry();
 int                   CheckRecentRREQ();
 int                   ClearAccRREQ_RERR();
 int                   CheckNeiList();
 int                   CheckLinkFailList();


   int         UpdateHello(struct HELLO_msg *);
   int         routing(u_long dst, Packet *p);
 int                   updateSimpleRRoute(u_long prevhop_ip);
 int             updateRT(u_long dst, u_long nexthop, u_int32_t seqno, u_int16_t hopcount, u_int64_t lifetime);

   int         sendRREQ(u_long dst, u_char ttl);
   int         forwardRREQ(struct RREQ_msg *my_rreq, u_char cur_ttl);
   int         sendRREP(u_long dst, u_long src, u_long toward, u_int8_t hopcount, u_int32_t seqno, u_int64_t lifetime);
   int         forwardRREP(struct RREP_msg *my_rrep, u_char cur_ttl);
   int         sendRERR(u_long delip, Unreach_list *);
   int         bcastRERR(Unreach_list *);

   int         push(void);
 int         sendToQueue(ePacket_*);
 int         processBuffered(u_long);
 int         LinkLayerCall(ePacket_ *);
 int         PrintIP(u_long);
 int         getnexthopAStar(u_long);  ///atanda
};

}; //namespace UserModule02d

#endif  /* __user_module_02_h__ */

```

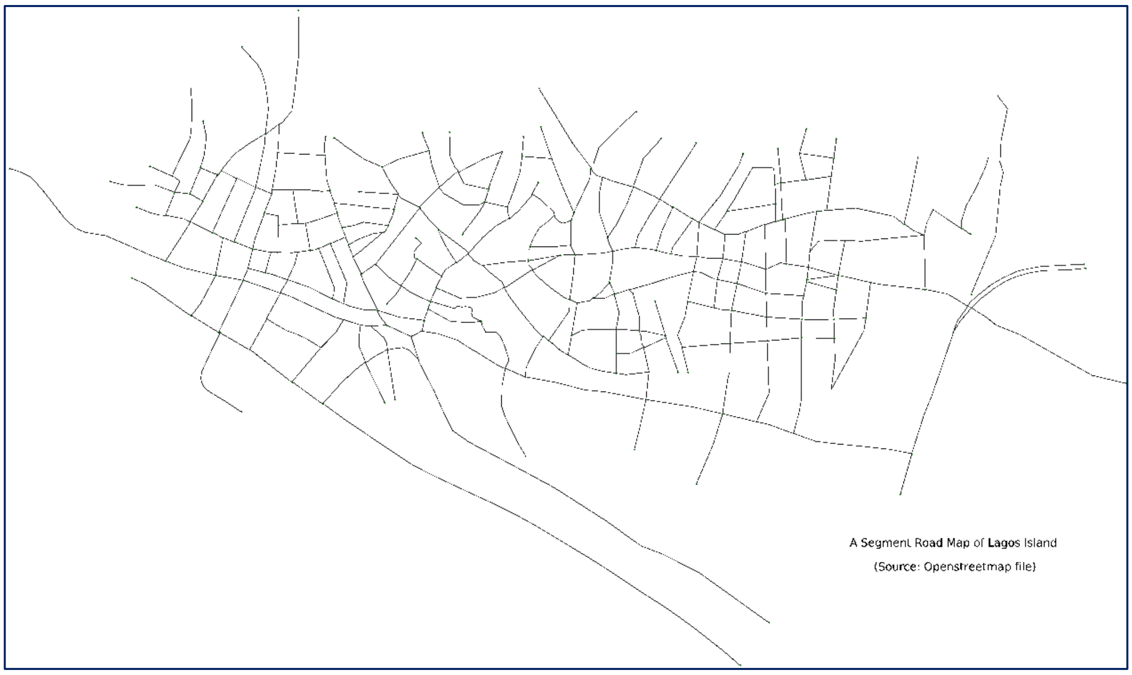
Appendix A3: Usermodule02.mdf file content sections that define user interface dialog box for the entry of parameters as shown in Fig. 3.9.

```
ModuleSection
    HeaderSection
        ModuleName          UserModule02
        ClassName           UserModule02

        NetType                     Wireless
        GroupName           User_Defined
        PortsNum            SinglePort

        Version             UserModule02_001
        Author              EstiNet
        CreateDate          2/26/02
        Introduction        "This is a UserModule02 module (Use this)."

        Parameter           HELLO_INTERVAL          1000    local
        Parameter           ALLOWED_HELLO_LOSS      2       local
        Parameter           ACTIVE_ROUTE_TIMEOUT    3000    local
        Parameter           DELETE_PERIOD           3000    local
        Parameter           NET_DIAMETER            15      local
        Parameter           NODE_TRAVERSAL_TIME     40      local
        Parameter           RREQ_RETRIES            5       local
        Parameter           RREQ_RATELIMIT          10      local
        Parameter           RERR_RATELIMIT          10      local
        Parameter           VDIST_MAX           1000    local
    EndHeaderSection



    End

        Begin TEXTLINE              VDIST_MAX
            Caption         "VDIST_MAX Limit      "
            Scale           10 288 200 30
            ActiveOn        MODE_EDIT
            Enabled                 TRUE

            Type            INT
            Comment                 ""
        End

        Begin LABEL         l_VDIST_MAX
        Caption        "(meters)"
        Scale       215 288 60 30
        ActiveOn      MODE_EDIT
            Enabled                 TRUE
    End

        Begin BUTTON                b_ok
            Caption         "OK"
            Scale           270 21 60 30
            ActiveOn        MODE_EDIT
```

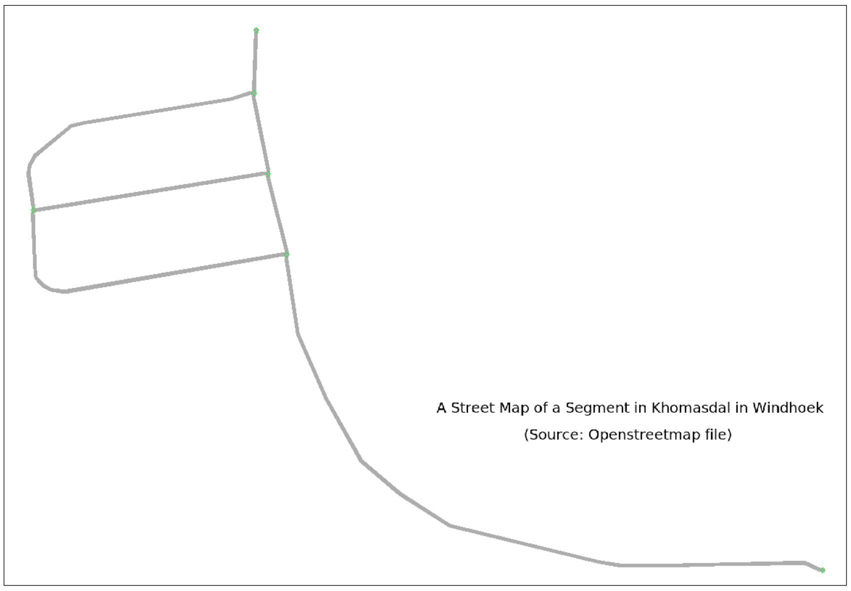
```
                Action          ok
                Comment                 "OK Button"
        End

        Begin BUTTON                    b_cancel
                Caption         "Cancel"
                Scale           270 53 60 30
                ActiveOn        MODE_EDIT
                Action          cancel
                Comment                 "Cancel Button"
        End
    EndInitVariableSection

    ExportSection
        Caption                 ""
        FrameSize               0 0
    EndExportSection
EndModuleSection
```

Appendix B: Street maps

Appendix B1: An Openstreetmaps segment figure of the Lagos Island; with dimensions of 125000m x 7500m.

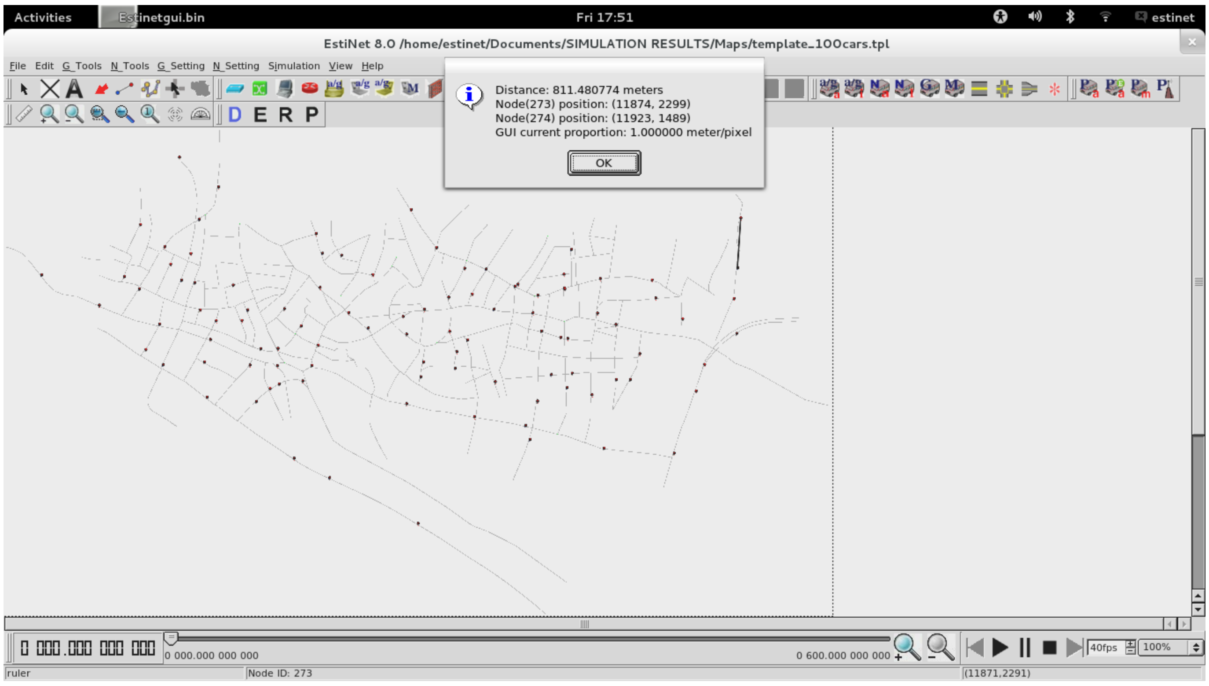

A Segment Road Map of Lagos Island
(Source: Openstreetmap file)

Appendix B2: An Openstreetmaps segment figure of a street in the city of Windhoek, with dimensions of 17000m x 1190m


A Street Map of a Segment in Khomasdal in Windhoek
(Source: Openstreetmap file)

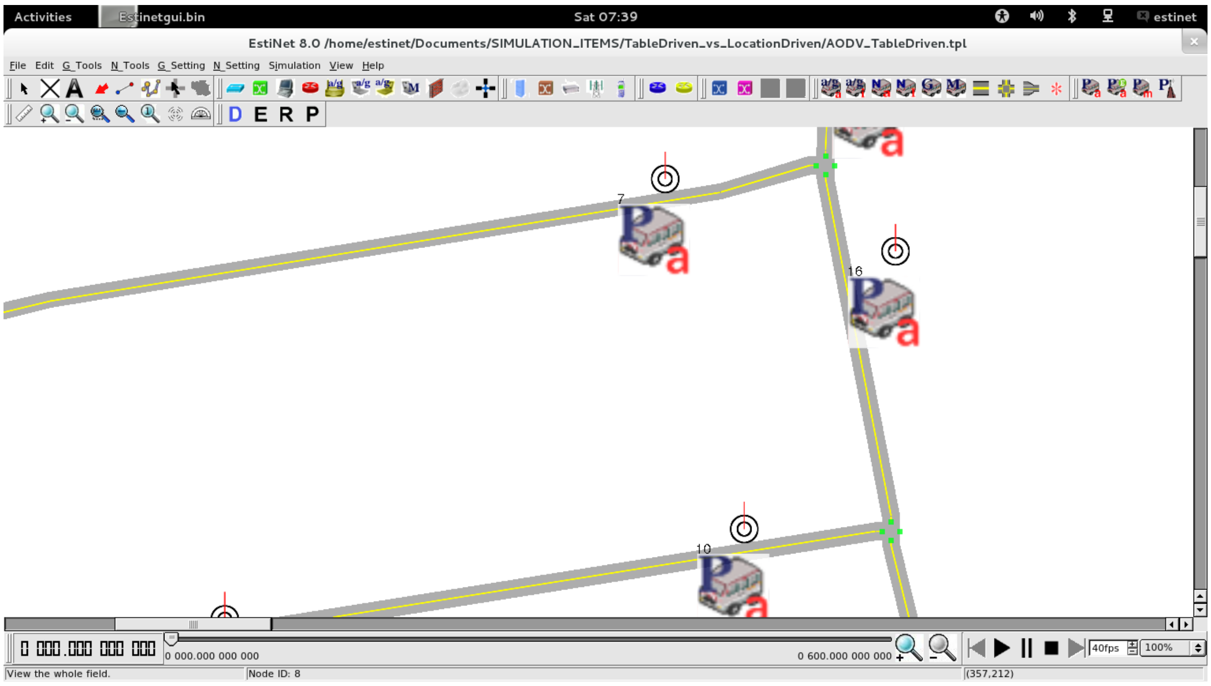
Appendix C: Node densities

Appendix C1: A VANET graph showing of 100-node 1/540000m^2 density. The short vertical line depicts a distance of 810m.


EstiNet 8.0 /home/estinet/Documents/SIMULATION RESULTS/Maps/template_100cars.tpl

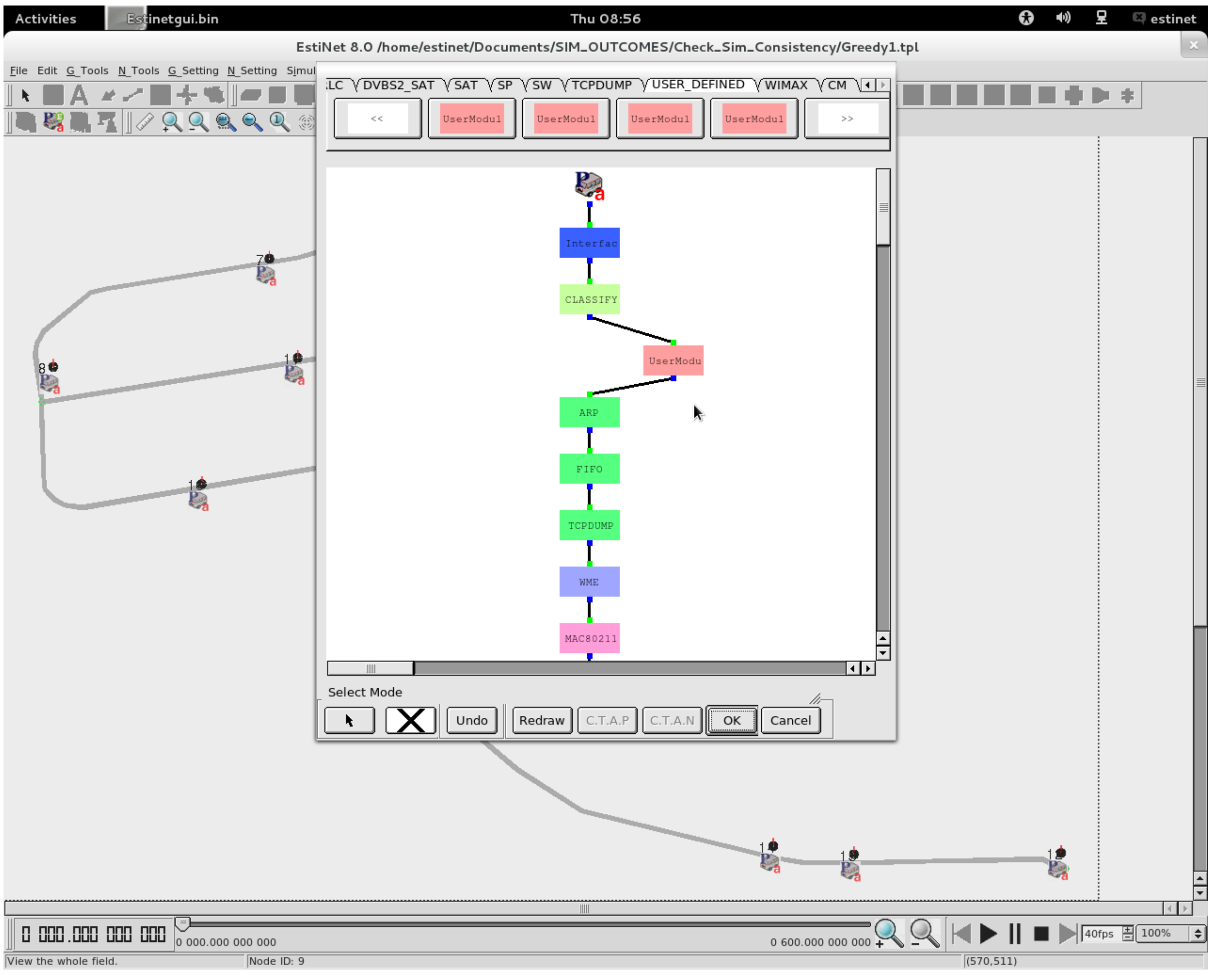
```
Distance: 811.480774 meters
Node(273) position: (11874, 2299)
Node(274) position: (11923, 1489)
GUI current proportion: 1.000000 meter/pixel
```

Appendix C2: A GUI close-up view of vehicular nodes in the EstiNet simulator


EstiNet 8.0 /home/estinet/Documents/SIMULATION_ITEMS/TableDriven_vs_LocationDriven/AODV_TableDriven.tpl

Appendix D: Networking protocols stack view in the EstiNet simulator

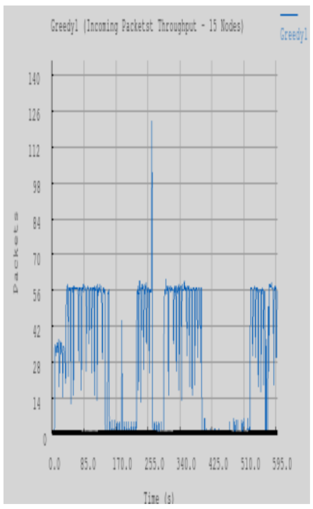
EstiNet simulator's protocol stack view, showing the insertion of a user developed routing protocol module (i.e. Greedy or S* routing module).

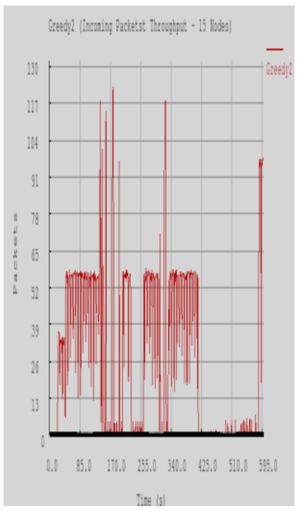


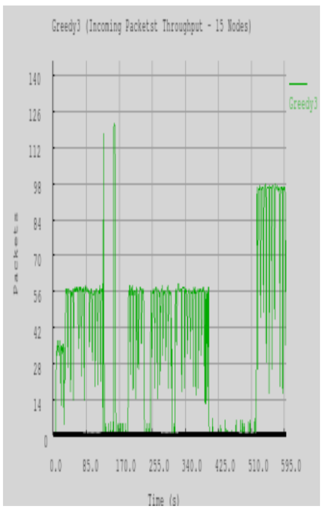
Activities
Estinetgui.bin
Thu 08:56
estinet
EstiNet 8.0 /home/estinet/Documents/SIM_OUTCOMES/Check_Sim_Consistency/Greedy1.tpl
File Edit G_Tools N_Tools G_Setting N_Setting Simul
LC DVBS2_SAT SAT SP SW TCPDUMP USER_DEFINED WIMAX CM
<< UserModul UserModul UserModul UserModul >>
Interfac
CLASSIFY
UserModu
ARP
FIFO
TCPDUMP
WME
MAC80211
Select Mode
Undo
Redraw
C.T.A.P
C.T.A.N
OK
Cancel
0 000.000 000 000
0 600.000 000 000
40fps
100%
View the whole field.
Node ID: 9
(570,511)

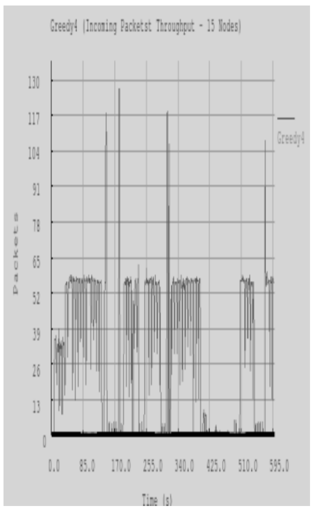
Appendix E: Confidence interval determinant

The following are different output graphs that result from the re-run of a 15-node topology VANET simulation file. The graphs exhibit a common packets delivery rate at level 56 in this case. We therefore deem the outputs of the simulator as being consistent.









Appendix F: A dense VANET scenario

Appendix F1: Dense traffic on a Lagos street

NMhttp://connectnigeria.com/articles/2012/08/14/lagos-traffic-the-new-laws-and-solutions/

Accessed 08/12/2014