Traffic Locality Oriented Route Discovery Algorithms for Mobile Ad Hoc Networks

A Thesis Submitted

By

Mznah A. Al-Rodhaan

For

The Degree of Doctor of Philosophy

To

The Faculty of Information and Mathematical Sciences
University of Glasgow

Abstract

There has been a growing interest in Mobile Ad hoc Networks (MANETs) motivated by the advances in wireless technology and the range of potential applications that might be realised with such technology. Due to the lack of an infrastructure and their dynamic nature, MANETs demand a new set of networking protocols to harness the full benefits of these versatile communication systems.

Great deals of research activities have been devoted to develop on-demand routing algorithms for MANETs. The route discovery processes used in most on-demand routing algorithms, such as the Dynamic Source Routing (DSR) and Ad hoc On-demand Distance Vector (AODV), rely on simple flooding as a broadcasting technique for route discovery. Although simple flooding is simple to implement, it dominates the routing overhead, leading to the well-known broadcast storm problem that results in packet congestion and excessive collisions. A number of routing techniques have been proposed to alleviate this problem, some of which aim to improve the route discovery process by restricting the broadcast of route request packets to only the essential part of the network. Ideally, a route discovery should stop when a receiving node reports a route to the required destination. However, this cannot be achieved efficiently without the use of external resources; such as GPS location devices.

In this thesis, a new locality-oriented route discovery approach is proposed and exploited to develop three new algorithms to improve the route discovery process in on-demand routing protocols. The proposal of our algorithms is motivated by the fact that various patterns of traffic locality occur quite naturally in MANETs since groups of nodes communicate frequently with each other to accomplish common tasks. Some of these algorithms manage to reduce end-to-end delay while incurring lower routing overhead compared to some of the existing algorithms such as simple flooding used in AODV. The three algorithms are based on a revised concept of traffic locality in MANETs which relies on identifying a dynamic zone around a source node where the zone radius depends on the distribution of the nodes with which that the source is “mostly” communicating.

Abstract

The traffic locality concept developed in this research form the basis of our Traffic Locality Route Discovery Approach (TLRDA) that aims to improve the routing discovery process in on-demand routing protocols. A neighbourhood region is generated for each active source node, containing "most" of its destinations, thus the whole network being divided into two non-overlapping regions, *neighbourhood* and *beyond-neighbourhood,* centred at the source node from that source node prospective. Route requests are processed normally in the neighbourhood region according to the routing algorithm used. However, outside this region various measures are taken to impede such broadcasts and, ultimately, stop them when they have outlived their usefulness. The approach is adaptive where the boundary of each source node's neighbourhood is continuously updated to reflect the communication behaviour of the source node.

TLRDA is the basis for the new three route discovery algorithms; notably: Traffic Locality Route Discovery Algorithm with Delay (TLRDA-D), Traffic Locality Route Discovery Algorithm with Chase (TLRDA-C), and Traffic Locality Expanding Ring Search (TL-ERS). In TLRDA-D, any route request that is currently travelling in its source node's beyond-neighbourhood region is deliberately delayed to give priority to unfulfilled route requests. In TLRDA-C, this approach is augmented by using chase packets to target the route requests associated with them after the requested route has been discovered. In TL-ERS, the search is conducted by covering three successive rings. The first ring covers the source node neighbourhood region and unsatisfied route requests in this ring trigger the generation of the second ring which is double that of the first. Otherwise, the third ring covers the whole network and the algorithm finally resorts to flooding.

Detailed performance evaluations are provided using both mathematical and simulation modelling to investigate the performance behaviour of the TLRDA-D, TLRDA-C, and TL-ERS algorithms and demonstrate their relative effectiveness against the existing approaches. Our results reveal that TLRDA-D and TLRDA-C manage to minimize end-to-end packet delays while TLRDA-C and TL-ERS exhibit low routing overhead. Moreover, the results indicate that equipping AODV with our new route discovery algorithms greatly enhance the performance of AODV in terms of end-to-end delay, routing overhead, and packet loss.

Acknowledgements

This thesis has been an inspiring, very challenging, but always interesting and exciting experience.

I am very grateful to my advisers, Dr. Lewis M. Mackenzie and Dr. Mohamed Ould-Khaoua, for their patience, encouragement, and many fruitful discussions from the very early stage of this research.

My gratitude also goes to Dr. Peter Dickman for his helpful comments in my second year report.

I am thankful for King Saud University, my employer, for providing me with the scholarship and the University of Glasgow for giving me the opportunity to join its computing science Ph.D. program.

Special thanks to the Computing Science Department at University of Glasgow for providing me with such wonderful research environment and facility needed to conduct my research. Also the Saudi Arabian Cultural Bureau in London for their support and facilitation.

Finally, I wish to express my appreciation to my family for their continuous love and encouragement, for always believing in me, for never failing to provide all the support, and for coping with the pressure that naturally comes with such endeavour.

Table of Contents

Table of Figures

Table of Tables

Acronyms

ABR	Associativity-Based Routing
ADR	Angle Deviation Ratio
AODV	Ad hoc On Demand Distance Vector
AOMDV	Ad-hoc On-demand Multipath Distance Vector
AP	Access Point
BCT	Broadcast Cache Time
B-ERS	Blocking-Expanding Ring Search
B-ERS+	Blocking-Expanding Ring Search Plus
CBR	Constant Bit Rate
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance
CTS	Clear-To-Send
DCF	Distributed Coordination Function
DSDV	Destination-Sequenced Distance-Vector
DSR	Dynamic Source Routing
DYMO	Dynamic MANET On-demand
ERS	Expanding Ring Search
FCFS	First Come First Served
FIFO	First In First Out
GPS	Global Positioning System
HoWL	Hop-Wise Limited broadcast
i.i.d	independent and identically distributed
ID	Identification
IEEE	Institute of Electrical and Electronics Engineers

LAR	Location Aided Routing
L-B	Limited Broadcasting
LHBA	Limited-Hop Broadcast Algorithm
LP	Locality Parameter
MAC	Medium Access Control
MANET	Mobile Ad hoc NETwork
MIMO	Multiple input multiple output
MP-DSR	Multi-Path Dynamic Source Routing
NETTT	NETwork Traversal Time
NTT	Node Traversal Time
OLSR	Optimized Link State Routing Protocol
P2P	Peer-to-peer
PANDA	Positional Attribute based on the Next-hop Determination Approach
PCF	Point Coordination Function
PDA	Personal digital assistant
r.v.	random variable
RF	Radio Frequency
RFC	Request For Comments
RPGM	Reference Point Group Mobility
RRD	Random Rebroadcast Delay
RRGM	Reference Region Group Mobility
RTS	Request-To-Send
RVGMM	Reference Velocity Group Mobility Model
RWP	Random Way Point

SDR	Speed Deviation Ratio
TCDS	Two-hop Connected Dominating Set
TL-ERS	Traffic Locality-Expanding Ring Search
TLRDA	Traffic Locality oriented Route Discovery Approach
TLRDA-C	Traffic Locality oriented Route Discovery Algorithm with Chase
TLRDA-D	Traffic Locality oriented Route Discovery Algorithm with Delay
TTL	Time To Live
UDP	User Datagram Protocol
VANET	Vehicular Ad-Hoc Network
VBR	Variable Bit Rate
Wi-Fi	Wireless Fidelity
WLAN	Wireless Local Area Network
ZHLS	Zone-based Hierarchical Link State
ZRP	Zone Routing Protocol

Chapter 1: Introduction

A computer network is a collection of independent devices that are interconnected together with the aid of some communication facilities. Until the early 1970s, computers were considered separately from communication. A decade later, wired networks were well established as a result of merging these two technologies [84]. Fixed networks are useful but not suitable for mobile situations. When mobile devices such as notebooks and personal digital assistant (PDAs) became widespread, this requirement generated intense interest in wireless networking. Modern wireless networks are: 1) *infrastructure* oriented such as a communication satellites [120] or a cellular network [68] 2) *infrastructure-less* such as Mobile Ad hoc NETwork (MANET) [25, 84, 117] or wireless mesh network [5]. A network can be wireless, mobile or both. A node in a wireless network can be stationary e.g. a desktop equipped with Wi-Fi, while a mobile node with a limited form of mobility can be part of a wired network. MANETs are both wireless and mobile.

One of the dominant initial motivations for MANET technology came from military applications in infrastructure-less environments [103]. However, while such applications remain important; MANETs' research has diversified into areas such as sensors networks, Vehicular Ad-Hoc Network (VANET) such as taxi cab network, civilian environments such as conference rooms or sports stadiums, emergency operations such as search and rescue operations or fire fighters, and personal area networks [93, 117].

In this chapter, wireless networks in general are examined briefly and MANETs in particular in greater detail, focusing on research trends, routing strategies, and classification. Afterwards, some of the most recently proposed MANETs routing techniques in the literature are discussed along with their approaches to route discovery. We then state the thesis statement and the main contributions made by this research. Finally, we provide an outline for the rest of the dissertation.

1.1 Wireless Networks

Nowadays, most mobile devices are equipped with short-range radio transmitters allowing them to inter-communicate using radio frequencies to transmit data and communicate with other devices

on the same network. Wireless LANs are standardised under the IEEE 802.11 series [55]. IEEE 802.11 standard defines two operational modes: infrastructure and infrastructure-less (known as the *ad hoc* mode). Infrastructure-oriented organisation is realised through fixed (typically wired) gateways or *access points* (*APs*) [68, 84, 117] that act as bridges to a fixed infrastructure. A mobile unit in such a network connects to the nearest AP which is within its communication range in a single-hop communication technique as depicted in Figure 1-1. The AP can connect other wireless nodes within its range with an existing wired network where the infrastructure mode is commonly used to construct a hotspot which provides a wireless access to the Internet. In the ad hoc mode, wireless nodes can communicate directly with each other. Infrastructure-less networks are commonly known as MANETs [84, 93] when they include mobile nodes. A MANET consists of a collection of spatially distributed nodes that communicate with each other over a wireless medium using multi-hop communication techniques without the need for fixed routers. Access to the Internet could be established with the help of nodes that are connected to the service thus these nodes act as gateways for the other nodes in the network.

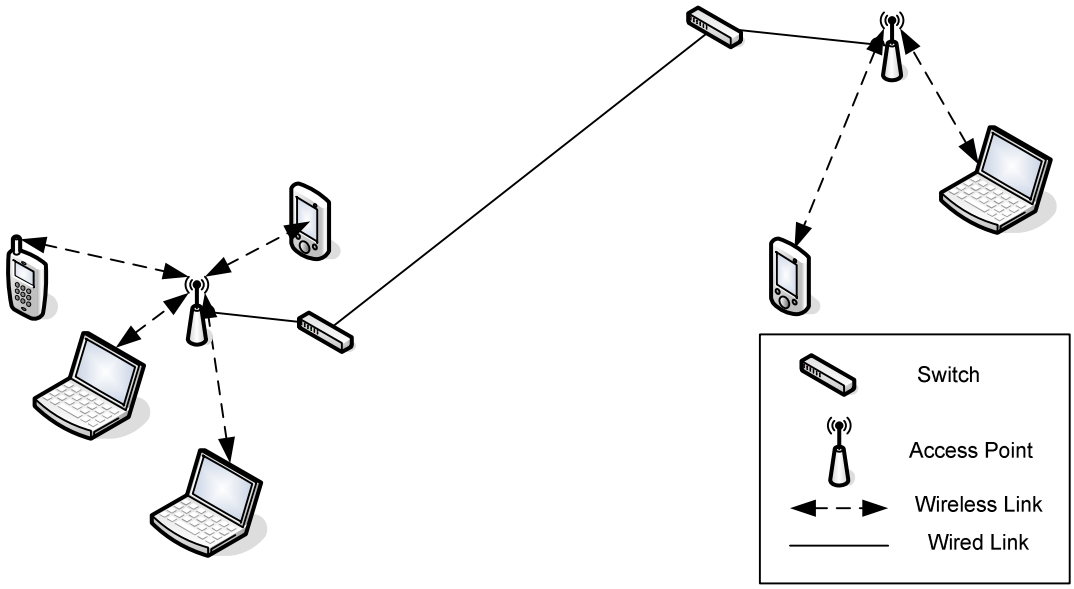

Figure 1-1: Infrastructure wireless network.

IEEE 802.11 [1] legacy is the standard for wireless local area network (WLAN) communication and has amendments such as 802.11a , 802.11b , and 802.11g [55], as well as 802.11n [91]. The IEEE 802.11 operates in the 2.4GHz band and supports data rates up to 2Mbps. However, 802.11a and 802.11g support a rate up to 54Mbps while 802.11b supports a rate up to 11Mbps. 802.11b and 802.11g operate in the same 2.4GHz band as the original standard while 802.11a operates in

the 5 GHz band. A detailed description of these extensions can be found in [7, 88, 114]. A new amendment 802.11n defines two 20 MHz bandwidth streams in both the 2.4 and 5 GHz bands and supports a rate up to 300Mbps. 802.11n uses new features such as a technology called multiple input, multiple output (MIMO) which uses several antennas to move multiple data streams [91].

1.2 MANETs

In MANETs, each node is equipped with a wireless transmitter and receiver and is typically free to move around in an arbitrary fashion. The self-configuration ability of MANETs makes them suitable for a wide variety of applications [25, 93] i.e. communication within groups of people through laptops and other hand-held devices. MANETs have gained a lot of attention from researchers around the globe over the past few years [2, 25, 81, 84, 104, 126].

MANETs require completely different protocols from those used for wired networks and infrastructure wireless networks [93]. This is because MANETs have their own constraints and require protocols that take into consideration mobility, bandwidth, and power consumption to provide the needed communication. Moreover, the fundamental challenge in MANETs is the design of functional spontaneous self-organised networks with low power, lightweight, and cheap components [45]. MANET characteristics differ from infrastructure networks since nodes can join and leave the network at any time. There is no central management and topologies change frequently and dynamically, so each node needs to act as a router to manage and provide routing facilities. This additional duty may consume network resources such as bandwidth and power.

1.2.1 Characteristics of MANETs

Due to the lack of fixed infrastructure, MANETs rely on wireless communication and collaboration among nodes as in Figure 1-2, introducing new challenging research issues related to routing, in particular where source and destination nodes rely on intermediate nodes to help in transmitting the packet to destination. This is because a node can only send data to another directly if they are within the transmission range of one another. Below we will briefly shed some light on some of these issues.

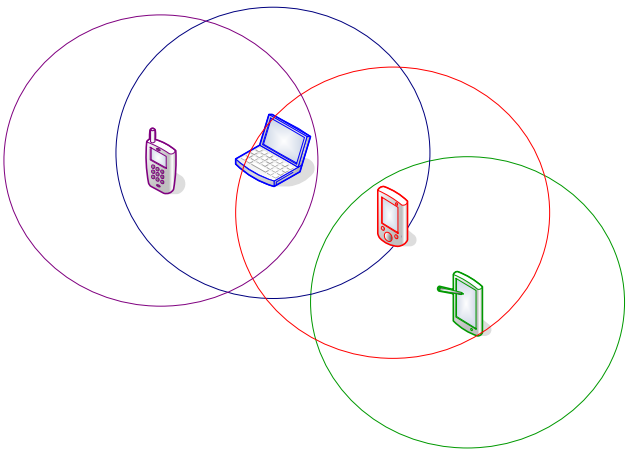

Figure 1-2: Connected MANET of 4 nodes with their transmission ranges depicted as circles.

Mobility:

As a mobile node moves, it may enter or leave the transmission ranges of other nodes. The establishment of routes depends on the relative location of the nodes and such routes may be repeatedly invalidated in an irregular and arbitrary fashion due to node mobility. Moreover, the mobility of a single node may affect several routes that pass through it. In fact, due to the nature of mobility a route that is considered active at a particular time may disappear and information concerning it become stale after a short time [19, 35].

In a MANET, the rate of topology change depends on the extent of mobility of an average node and its transmission range [77]. These multi-hop topologies may change randomly and rapidly in unpredictable fashion [84] because they are also highly influenced by nodes characteristics. Therefore, any node may disappear from the topology due to mobility, battery drainage, or simply being switched off. Meanwhile, nodes maintain their own logical identifiers and most of their resources as they move around.

Bandwidth Constraints:

MANETs have significantly lower communication capacity than traditional wired networks due to the fact that wireless links have limited bandwidth capacity [126]. This bandwidth limitation has been the focus of a great deal of research work aimed at alleviating the constraints placed on many applications [93]. In fact, the need for high bandwidth is expected to continue to increase as the applications get more sophisticated.

Power Constraints:

The mobile nodes in MANETs typically have good portability and flexibility. However, in many cases they are equipped with limited capacity power sources and are heavily constrained by battery lifetime [25]. In some scenarios, a node may exhaust its power supply where a replacement of power resources might be impossible. In many applications therefore, power conservation is a key aim; however, increasing the power dedicated to radio transmission and reception can broaden the radio range improving connectivity and boosting network functionality. Clearly, there is often a trade-off between the connectivity needed and the amount of energy consumed. Researchers have put considerable effort into the design of power-aware protocols [26].

1.2.2 Routing in MANETs

Routing protocols are invoked when a source node needs to send a packet to a particular destination. Due to the lack of infrastructure, routing algorithms used in MANETs differ from their counterparts used in other networks [56, 84, 93, 117]. The design of an efficient and reliable routing strategy is a very challenging problem due to the limited resources available so each intermediate node along the path from source to destination acts as a router, as shown in Figure 1-3. Many multi-hop routing protocols have been proposed and investigated in the literature [3, 49, 60, 93, 94, 130]. The routing protocols can be divided broadly into three categories [2, 84]: *proactive*, *reactive*, and *hybrid* based on the routing information update mechanism.

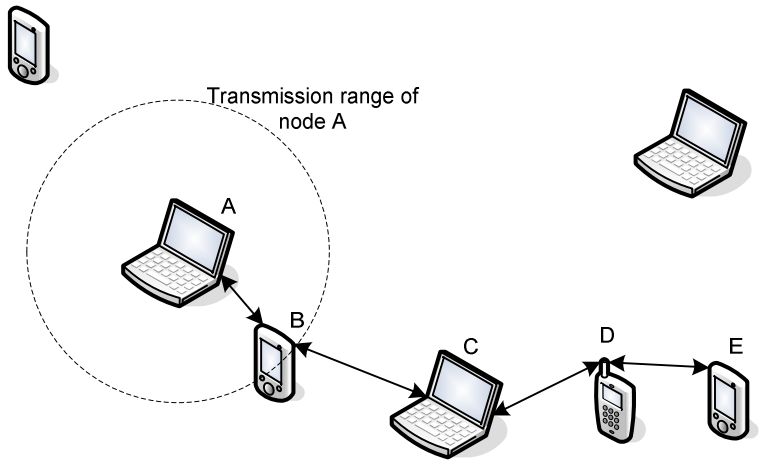

Figure 1-3: Routing in MANET through relaying from node A to node E.

In proactive routing protocols (table-driven), the routes to all the destinations are determined at the

start up and maintained by using periodical tasks to inform all nodes about routes status. So nodes maintain topology information in their routing tables collected from the periodically exchanged information which is flooded to the whole network. Any required route will be found from the routing table within the node. Examples of this class of routing protocols are the Optimized Link State Routing Protocol (OLSR) [3] and Destination-Sequenced Distance-Vector routing (DSDV) [95].

In reactive routing protocols (also known as *on-demand* protocols), routes are determined dynamically when required by a source node using a route discovery process. Its routing overhead is lower than the proactive routing protocols if the network size is relatively small [32]. An on-demand routing protocol has two phases.

- The *route discovery* phase is used to discover one or more routes leading to a particular destination. It is achieved using broadcasting techniques.
- The *route maintenance* phase is used to maintain the route by monitoring its operation within the network and informing other nodes of any routing errors or intermediate link failures.

Examples of this class are the Dynamic Source Routing (DSR) [60], Ad hoc On-demand Distance Vector (AODV) [94] and recently the Dynamic MANET On-demand (DYMO) [22] routing protocol.

Finally, hybrid routing protocols combine the basic properties of the above two classes of protocols. The Zone Routing Protocol (ZRP) [49] and zone-based hierarchical link state (ZHLS) [59] are examples where the network is divided into areas called *zones* where a proactive routing protocol operates inside each zone and a reactive protocol between zones.

On-demand routing protocols search for the desired route only when needed and avoid the use of periodical control packets for routing purposes to utilise bandwidth and power which makes the concept appealing for MANET scenarios [2, 33]. When a source node needs to send packets to an unknown destination, it initiates a *route discovery* process to look for one or more routes, as a backup, to that destination using broadcasting techniques. Once such a route is discovered, the

source node starts transmitting data packets. In on-demand routing protocols, the route discovery phase dominates most of the routing overhead and delays the data transmission. This research concentrates on the route discovery phase in on-demand routing algorithms in an effort to improve performance through the reduction of the route request overhead and route discovery time.

In MANETS, *broadcasting* is an essential part of routing. In on-demand routing protocols, it is used to discover a route or multiple routes. For example, both DSR and AODV use simple flooding as a means of broadcasting, where each node may receive multiple copies of a unique route request packet and retransmit it exactly once. Unfortunately, as is well known, flooding leads to packet redundancy that can cause congestion increase and packet collisions in the network. This phenomenon is widely known as the *broadcast storm problem* [131]. Moreover, flooding is wasteful of node resources such as power and bandwidth. The deleterious impact of this problem can be reduced if the broadcasting is controlled, for example by pruning the dissemination of the route request as soon as possible upon the discovery of the needed route [44, 89, 115].

In simple flooding, used in most existing on-demand protocols, when a source node needs to find a route to a particular destination, it first searches in its routing table where any discovered route is stored for future use; if this is unsuccessful, a new route discovery process is started whereby a route request packet is broadcast from node to node until it arrives at the destination, or an intermediate relay node that has a route to the destination. However, other nodes will continue to broadcast it until the time to live (TTL) field reaches zero.

In MANETs, as in wired networks, the TTL field limits a datagram's lifetime and is used to prevent packets from persisting in the network [50]. In practice it is in fact a hop count initialised by the source to a predefined initial value. Each intermediate routing node that a packet crosses decrements the TTL field by one until it hits zero whereupon the packet is discarded.

1.2.3 Improvements of the route discovery process

The route discovery process often floods the network with route request packets looking for a specific route throughout the network. Unfortunately, a given route request often keeps propagating even after the route has been found thus congesting the network and wasting

resources. Route discovery protocols could be improved by minimising such overhead and reducing or stopping the unnecessary propagation of route request packets after the route has been discovered. A number of approaches have been proposed to reduce this overhead by using limited variations of broadcasting; examples can be found in [38, 60, 79, 115, 124, 130-132].

Typically, a route request packet contains a TTL value that specifies the number of re-broadcasts allowed for that route request. So, the broadcast of the route request can be controlled using the TTL field. Expanding Ring Search (ERS) [61, 115] searches for the target in a multi-ring rather than a one-to-all scheme. This is achieved by performing several search attempts as rings by increasing the TTL value for each successive ring. The TTL value is increased from an initial value, when used as the radius for the first ring, by a fixed amount until it reaches a predefined threshold to expand the radius of the search linearly. The authors in [125] have found that the pessimistic search provides the best performance because the initial ring is bigger enough to include the needed route. Moreover, Hop-Wise Limited broadcast (HoWL) [78] is another approach that limits the dissemination the route request by predicting the destination node's location from old routes to that destination. Such approach does not always outperform ERS because the historical data have a higher chance of been stale information which will result in poor performance especially in high mobility environments.

An algorithm for route discovery optimisation that eliminates the need for historical or location information has been proposed in [44]. It achieves this by employing *chase packets* which are control packets that are broadcast after a route to the desired destination has been found, to stop the (now fulfilled) route request from further propagation. Chase packets are discarded upon the success of their mission. Limited-Hop Broadcast Algorithm (LHBA) [132] uses the chase packet technique from [44] where the chase packets are broadcast by route finders to a predefined number of hops to free this part of the network from the fulfilled route request. Blocking-Expanding Ring Search [89] is another algorithm that aims to improve energy consumption by introducing a delay that is equal to twice the hop-count at each node. After this delay if the chase packet has been received, the intermediate node discards the route request. Otherwise it rebroadcasts the route request to its neighbours.

1.3 Motivations and objectives

The concept of *locality* is central to many processes in life where it manifests itself in terms of time, activity, and space. Locality is frequently observed in computing systems, for example in program execution and storage management. In networking, locality of reference [65] is observed through the fact that nodes in the same geographical area tend to receive communication from the same source highlighting *spatial locality*. On the other hand, entities that have communicated within the near past have high probability of re-communicating again in the near future leading to *temporal locality*. Certain MANET applications may exhibit traffic behaviour that follows particular patterns in which the source node tends to communicate with certain set of nodes more than others regardless of their locations or time of communication, we call this traffic locality. An application might exhibit a combination of spatial, temporal or traffic localities. This observation has motivated here a new approach to traffic locality in MANETs based on the working sets concept that is widely adopted in memory and storage management [112] where the term "working set" refers to the collection of pages that a process is actively referencing in a given time period and therefore on which it tends to concentrate memory references. The working set can be introduced in MANET as the set of nodes that the source node is mostly communicating with (not necessarily direct neighbours) more than others. This set is not fixed and always updated to reflect the current communications.

On-demand routing protocols generally have a low overhead compared to the proactive protocols that use periodical control packets [33]. However, it is still desirable to further reduce the overhead as much as possible. Limiting the broadcast of the route request has the potential to reduce routing overhead and congestion level caused by the routing algorithm because the route discovery process dominates most of the overhead.

In this research, we will propose *adaptive* algorithms that improve the performance of the route discovery process for on-demand routing protocols. In these algorithms, each source node maintains a "neighbourhood region" containing most of the destinations that this node mostly communicates with enabling the algorithms to be adaptive. Moreover, this ability improves with time as they learn about the status of the network and adjust neighbourhood boundaries accordingly. We will use the concept of traffic locality as the base for the development of three

new algorithms: Traffic Locality Route Discovery Algorithm with Delay (TLRDA-D), Traffic Locality Route Discovery Algorithm with Chase (TLRDA-C), and Traffic Locality Expanding Ring Search (TL-ERS).

1.4 Thesis Statement

In MANETs, the route discovery process is an essential part of on-demand routing protocols and usually relies on simple flooding as a broadcasting mechanism to disseminate route requests. Unfortunately, simple flooding is expensive and leads to the *broadcast storm problem*. Performance can be improved if appropriate measures are taken to stem route request dissemination. One approach to such measures relies on the observation that, in many practical scenarios, network traffic exhibits some kind of locality where each source node tends to communicate with a certain subset of nodes more than others. Such a subset forms that source node's *neighbourhood region*.

In this thesis, I make the following assertions.

T1: Route requests should propagate as fast as possible prior to the discovery of the route to minimise the route discovery time. Our new Traffic Locality oriented Route Discovery Approach (TLRDA) divides the network into a *neighbourhood* and *beyond-neighbourhood* region for each prospective source node in applications that exhibit traffic locality in MANETs. In TLRDA, route requests propagate as fast as possible within their source node's neighbourhood region to avoid delaying the route discovery process.

T2: Unfulfilled route requests should always be given priority over fulfilled route requests. The new algorithm, TLRDA-D, uses the neighbourhood approach as stated in T1 and adds a deliberate additional delay to route requests that are broadcast in their source node's beyond-neighbourhood region. Adding a delay to the route requests propagation within their source node's *beyond-neighbourhood* region gives priority to other route requests that are propagating within their own source node's neighbourhood regions. Such a priority gives the route requests a chance to discover destinations earlier and reduces channel contention leading to improvement in the end-to-end delay. TLRDA-D improves the end-to-end delay as it speeds up the propagation of route requests that are broadcast within their own source node's neighbourhood region.

T3: Removing the fulfilled route requests from the network reduces routing overhead. To this end, the TLRDA-C algorithm uses *chase packets* to stop the fulfilled route requests reducing route request overhead and improving network performance without delaying the discovery process. TLRDA-C improves the routing overhead while showing the same improvement of the end-to-end delay as TLRDA-D.

T4: ERS reduces the routing overhead without increasing the end-to-end delay only if it succeeds in finding the needed route in the first ring. The suggested TL-ERS algorithm improves the existing ERS algorithm by employing the neighbourhood approach, introduced in T1, to increase the success in finding the route within the first ring. Since the neighbourhood region includes most of the destinations for the source node, the route discovery algorithm has a very high chance of finding the destination in the neighbourhood region from the first attempt, reducing the route request overhead without increasing the end-to-end delay. The maximum number of rings is kept low to improve network performance in the worst-case scenarios. TL-ERS reduces routing overhead and improves the end-to-end delay compared to ERS.

1.5 Contributions

In this research, we propose new algorithms to improve the performance of the route discovery process of on-demand routing protocols. These algorithms are *adaptive* in that they adjust each source node neighbourhood boundary according to the current situation to improve performance. There are four main contributions as stated above. Following, a brief summary of these contributions:

Traffic locality oriented Route Discovery Approach (TLRDA)

In TLRDA, a neighbourhood region is established for each particular source node that includes the most likely destinations. Nodes broadcast the route request without adding any extra delay within the route request source node's neighbourhood region in an effort to improve the route discovery process in applications that exhibit traffic locality for MANETs. This concept is the base for the development of the other algorithms suggested in this research.

Traffic Locality oriented Route Discovery Algorithm with Delay (TLRDA-D)

TLRDA-D utilises TLRDA to establish the neighbourhood and beyond-neighbourhood regions for each active source node. Nodes broadcast the route request without adding any delay while it is propagating within its source node's neighbourhood region. However, beyond this region the route request is further broadcast with a deliberate additional delay until such broadcast fades away as TTL reaches zero or the connected network is fully covered. The reason for adding this delay is to give priority to route requests that are travelling within their own source node's neighbourhood region since other route requests that are travelling in their source node's beyond-neighbourhood region have higher chance of being already fulfilled. One of the main advantages of TLRDA-D is improving the end-to-end delay because it does not hinder route requests that are broadcast within their own source node's neighbourhood region. This approach improves route discovery as well as the congestion level, by reducing channel contention throughout the network.

Traffic Locality oriented Route Discovery Algorithm with Chase (TLRDA-C)

TLRDA-C is a new route discovery algorithm that utilises the chase packet concept with TLRDA-D. Upon receiving a route reply, the source node transmits a chase packet to catch and terminate the original route request. The chase packet travels at full speed to terminate the propagation of the fulfilled route request not far beyond its neighbourhood region since the chase packet travels faster than the route request in the beyond-neighbourhood region; the route request is subject to a slight delay while propagating in this region. TLRDA-C minimises the overhead and reduces the end-to-end delay compared to Limited Broadcasting [44] Blocking-ERS [89] and simple flooding used in AODV [94].

Traffic Locality-Expanding Ring Search (TL-ERS)

This algorithm is an improvement to the Expanding Ring Search. It first broadcasts route requests using the neighbourhood region as a first locale or ring, in which to search for the target. If route discovery in this ring proves unsuccessful, the algorithm then establishes a second ring, double the size of the first, if route discovery here also fails the algorithm finally resorts to flooding. In both ERS and TL-ERS, there is a trade-off between network overhead and end-to-end delay.

TLRDA-D and TLRDA-C are found to be suitable for time sensitive applications such as instant messaging applications while TLRDA-C is for applications that are both time and overhead sensitive such as fire fighters working in teams. However, TL-ERS is suitable for overhead sensitive applications such as groups of college students exchanging email messages.

1.6 Thesis outline

The rest of this thesis is organised as follows:

- Chapter 2 provides an overview of the related work and background information which are necessary for the subsequent chapters. This chapter also provides the preliminaries for the mathematical and simulation models used to assess the performance of the new algorithms presented in the subsequent chapters. It starts with brief introduction of on-demand routing algorithms taking AODV as an example followed by an overview of the broadcasting approaches. After that, it describes related route discovery optimisation techniques. Finally, it reviews the notation, justification of the methods, simulation environment, assumptions, parameters, and metrics.

- Chapter 3 is devoted to the development of the traffic locality concept in MANETs and utilising this concept to improve the route discovery process through the development of Traffic Locality oriented Route Discovery Approach (TLRDA).

- Chapter 4 presents the Traffic Locality oriented Route Discovery Algorithm with Delay (TLRDA-D) that utilises the TLRDA using delay within the beyond-neighbourhood region to give priority to other route requests travelling within their source node's neighbourhood region. Queuing theory and simulation are used to conduct in depth investigation of its performance.

- Chapter 5 proposes the Traffic Locality oriented Route Discovery Algorithm with Chase packets, TLRDA-C, that utilises TLRDA and TLRDA-D, introduced in Chapter 3 and Chapter 4, in addition to the chase packet concept. It shows how exploring the concept of traffic locality with other techniques can help to reduce route request overhead and route request latency whilst keeping the same improvement of discovery time as TLRDA-D. It also presents a comparative performance study of our newly proposed algorithms:

TLRDA-C and TLRDA-D.

- Chapter 6 develops the Traffic Locality Expanding Ring Search, TL-ERS, as an improvement to Expanding Ring Search (ERS) in applications that exhibit traffic locality for MANETs. Also it presents a comparative performance study of our newly proposed algorithms: TL-ERS, TLRDA-C, and TLRDA-D.
- Chapter 7 concludes this thesis by summarising the main results and then outlines some possible directions for future work.

Chapter 2: Related Work and Preliminaries

2.1 Introduction

In MANETs, the design of an efficient routing protocol that can cope with the system's constraints, such as mobility, bandwidth, and power is a very challenging task [84]. Applying routing protocols that were designed to work in wired networks to resources-sensitive environments such as MANETs without proper modifications is impractical [56, 84, 93, 117]. As a result, various routing algorithms have been proposed for MANETs over the past years [3, 49, 60, 94, 130].

Broadcasting is used in many MANETs applications [18, 122] and is an essential operation of many routing protocols in that it is used to discover new routes between source and destination pairs. In MANETs, conventional flooding is simple but costly [131]. Broadcasting in MANETs has been the subject of intensive research [6, 8, 20, 24, 44, 71, 79]. Controlling the broadcast of route requests to cover part of the network [20, 23, 38, 44, 66, 89], at least initially, as opposed to unrestricted network coverage can help to alleviate such effects and improves network performance in terms of overhead and congestion levels.

In this chapter, we will introduce related work that has been presented in the literature then establish some necessary preliminaries and notation that will be used throughout the rest of this dissertation.

2.2 Related Work

This section first describes the traditional on-demand routing protocol for MANETs; namely, the Ad hoc On-demand Distance Vector (AODV) protocol [93, 94] that uses simple flooding. It is one of the well-known routing protocols that has been widely investigated in the literature [2, 84, 93, 104, 117]. Due to its popularity, it will be used throughout this study for comparisons and benchmarking purposes. As in all on-demand routing protocols, the operation of AODV protocol consists of two phases: *route discovery* and *route maintenance*.

When a source node needs to send data to a destination, but does not have a valid route to that destination, it initiates the route discovery phase to find a valid route. The source node broadcasts a route request packet to its neighbours which in turn forward the route request packet to their neighbours and so on. Each node that forwards a route request creates a reverse route back to the source node itself. The route request packet is broadcast until it either reaches the destination or a node which contains a fresh route to the destination in its cache; the finder can be either the destination or an intermediate node. Once a fresh route is found, the finder node transmits a unicast route reply packet to the source node using the reverse route. Each node that participates in forwarding the route reply back to the source creates a forward route to the destination, storing a pointer to the next hop neighbour rather than storing the entire path. AODV uses *sequence numbers* to ensure that routes are loop-free and fresh to avoid stale information; and each node maintains its own sequence number [94]. The source node includes its own sequence number and the most recent destination sequence number in the route request packet. Intermediate nodes reply to the route request query only if they have fresh routes to the destination where the fresh route sequence number is greater or equal to the one contained in the route request packet.

The route maintenance phase is triggered when a node detects a broken link. The node that has detected the broken link sends a route error packet to the neighbours that are actively using the route; to inform them about the invalid route. For this purpose, AODV uses an active neighbour list to keep track of first-hop neighbours that are using a particular route. The node also removes the routing entry from its table. This procedure is repeated by all nodes that receive the packet. The source node may request a new route by broadcasting a new route request if it has more data packets to send.

2.2.1 Broadcasting in on-demand routing protocols

Broadcasting is a crucial communication operation in MANETs and an essential part of most of its routing protocols [2, 8, 72, 104, 131]. Broadcasting can be classified as deterministic and probabilistic [119, 123]. The former can guarantee complete coverage depending on the node distribution while the latter may not because its coverage depends also on the choice of the probability for forwarding route requests.

The deterministic approach guarantees full coverage in a connected network so it is reliable; a basic example is simple flooding [29, 93, 98]. In more sophisticated examples, a node uses information gathered from its neighbours to limit the forwarding of a broadcast packet in an attempt to reduce redundant transmissions. For instance, in Self Pruning [30, 72, 124], intermediate node rebroadcasts only if it can reach additional nodes. While in Dominant Pruning [72], each node chooses some or all of its 2-hop neighbours as rebroadcast nodes. In MultiPoint Relaying (MPR) [100] each node selects a set of its neighbours as its MPRs so that all its 2-hop neighbours can be reached through its MPR set. Two-hop Connected Dominating Set (TCDS) [113] takes into account three-hop information to select the relay nodes for broadcasting.

The probabilistic approach is simple but unreliable because each node broadcasts according to a predetermined probability depending on specific criteria [74, 101]. This is achieved by inhibiting some intermediate nodes from forwarding the received packets using some local topological characteristics. However, the network coverage is increased with the increment of the probability factor in a connected network. Examples of the probabilistic approach include Counter-based and Distance-based methods [116, 131]. In the Counter-based scheme a node rebroadcasts a packet only if it receives fewer redundant copies than a predefined threshold within a random time length. In Distance-based scheme, a node rebroadcasts a packet only if the shortest distance to its nearest neighbour who sent a redundant copy is greater than a predefined threshold within a random time interval; the distance is measured by the signal strength.

Most popular on-demand routing algorithms such as AODV [94] and DSR [60] use simple flooding to discover new routes due to its simplicity [20, 34, 123]. When a source node needs a route for a given destination, it broadcasts a route request packet to all reachable nodes in the network with the help of intermediate nodes as relays. Each intermediate node participates in delivering the route request by broadcasting it only once and discards all redundant packets blindly after that. Flooding consumes lots of resources such as bandwidth and power and this is the cause of the broadcast storm problem [62, 116, 131], In fact a broadcast storm is a combination of three sub-problems:

- *redundancy*, a node might receive many copies of the same packet;

- *contention*, a node tries to broadcast but get delayed because a neighbouring node is using the shared media;
- *collision*, two neighbouring nodes start transmitting simultaneously and the two packets collide with each other.

In on-demand routing protocols, the broadcasting of route requests used in the route discovery process dominates most of the routing overhead when relying on simple flooding as a form of broadcasting [92, 115, 122]. Several approaches have been proposed to reduce this overhead by using a controlled variation of broadcasting whether it is flood-based or not as in [20, 24, 30, 39, 44, 89, 102, 131, 132].

Several methods have been suggested to alleviate the broadcast storm problem associated with flooding. Algorithms based on probabilistic broadcast mitigate the broadcast storm problem by reducing the number of redundant packets to reduce network congestion. However, this problem can be eased by preventing broadcast synchronisation between neighbouring nodes because when they sense an idle channel, they may start sending at the same time resulting in a collision. This prevention could be achieved by introducing a jitter uniformly distributed between 0 and 10 ms [18] before broadcasting at each node; the jitter is known later as Random Rebroadcast Delay (RRD) [70, 110]. The Positional Attribute based on the Next-hop Determination Approach (PANDA) [71] uses location, velocity or power information at each relay node to set up RRD so that a better candidate rebroadcasts first giving it a priority over other instances of the rebroadcast.

2.2.2 Improvements to route discovery process

The route discovery process can be improved by controlling the route request dissemination to avoid unnecessary network coverage. Methods for improving the route discovery process which are not flood-based can be categorised by the method used for controlling the broadcasting, i.e. Time-To-live (TTL), *chase* packets, location, and neighbour.

Improvements using TTL

The broadcast of the route request can be controlled using the TTL field in the route request packet. Expanding Ring Search (ERS) is one of the route request improvement techniques to incur

lower overhead where source node searches for the target in multi rings scheme instead of one-to-all scheme where each ring is centred at the source node. ERS was adopted first in DSR [61] using a two-ring scheme where the route request is broadcast to cover the first hop neighbour by broadcasting the packet with TTL equal 1; if unsuccessful simple flooding is used. Later ERS was proposed for AODV [115] but with a different mechanism which uses a multi-ring scheme. This is achieved by increasing the TTL value, by a fixed amount, at each ring to expand the radius of the search linearly which may increase the end-to-end delay. More details about ERS can be found in Section 6.1.1.

Researchers in [125] have tried to find the best initial value for TTL theoretically where each source node can estimate the distance to the destination assuming that the destination node's mobility speed is available to the source node. They have found that the pessimistic search, where the initial ring contains the required route provides the best performance. A study in [23] has proposed two approaches: the first assumes the probability distribution of the destination is known prior to the discovery process, and the second assumes such a distribution is not known. The latter reflects more realistically the unpredictability of MANETs and uses a sequence of random TTL values to minimise the worst-case search cost. It has been further investigated in [64] while caching of previous routes is taking into consideration. They found out that this approach has similar overhead but higher delay compared to the basic route discovery in DSR.

Hop-Wise Limited broadcast (HoWL) [78] is another approach that limits the route request by predicting the destination location from old routes. It sends the route request packet with a TTL equal to the average of hop counts of all old stale routes to that particular destination plus a constant value if the destination is known to the source node; otherwise it uses the simple flooding.

Improvements using chase packets

A chase packet is a control packet that is broadcast after finding the desired route to stop a fulfilled route request from further propagation. Limited Broadcasting is an algorithm proposed in [44] for route discovery process that eliminates the need for historical or location information. It achieves this by employing chase packets to control the propagation of the fulfilled route requests. When the distance between the source and the destination nodes is unknown in a bidirectional network,

nodes broadcast route requests using only ¼ of the channel time to slowdown the route requests' propagation while the rest of the channel time is used to transmit route replies and broadcast chase packets such that chase packets are three times faster than route requests to give the chase packets a chance to catch the fulfilled route requests. This technique will delay route requests and route replies increasing the end-to-end delay. Limited Broadcasting will be explained in detail later in Section 5.1. Moreover, in this algorithm the sender is solely responsible for initiating the chase packet which might experience an extra delay in catching the route request. This shortcoming of Limited Broadcasting has been addressed in the Limited-Hop Broadcast Algorithm (LHBA) proposed in [132]. LHBA allows any node that discovers a route to initiate a chase packet. The chase packet is broadcast by the route finders to K hop neighbours to free this part of the network from the fulfilled route request. However, this algorithm may congest the network by generating many chase packets when trying to stop the same route request which may cause a storm of chase packets.

Blocking-ERS [89] is another algorithm that aims to improve the energy consumption by controlling route request dissemination. This algorithm uses chase packets to improve the route request process. It works by introducing a delay equal to twice the hop-count at each node and before discovering the route which may increase the end-to-end delay. After this delay the intermediate node may receive a chase packet called "stop_instruction" from the source node to cover up to the ring where the finder of the needed route resides which may reduce the success rate of the catching process in mobile situations. Upon receiving the chase packet, the intermediate node discards the route request. If the chase packet is not received, the intermediate node rebroadcasts the route request to cover a larger area. A detailed explanation of Blocking-ERS is presented in Section 5.1.

Improvements using location information

Some researchers [71, 79, 130] have tried to reduce the overhead of the discovery process with the aid of location or distance information. The Location Aided Routing (LAR) protocol [130] limits the search for a new route to a limited zone called the *requested zone* through the aid of the Global Positioning System (GPS). In such a protocol the broadcasting overhead is reduced but the location information may not be available in some scenarios due to unavailability of GPS or the

weakness of its signal in indoor situations. Recently, a study in [66] has proposed a novel approach to adjust route discoveries dynamically in LAR by combining it with the Distance-based scheme.

Selective Flooding (SF) [79] limits the broadcast of route requests to a selected area. This selection is based on the hop-counts for the destination which are stored in a source distance table within each node. Nodes in SF must receive periodic packets within a short time interval in order for this algorithm to work properly.

Improvements using neighbour information

The broadcast can be limited by using previously cached historical route information. The algorithm discussed in [20] broadcasts to a small region defined by prior routes that have been stored inside each node. The algorithm reduces the route discovery overhead by reducing the region to be flooded depending on this information but it has a high chance of being stale. However, it has been improved in [38] by storing the encounter time of the destination to select the most recent route rather than the first route found in the cache; both algorithms pay a high price in scalable networks, where the network size maybe in thousands , because they require the storage of large amount of historical data which consume memory and power.

2.3 Preliminaries

In this section, the necessary preliminaries used throughout this research are presented. We introduce the notation used in the subsequent chapters, provide a justification of the methods used in the performance analysis, and then describe the simulation environment, assumptions, parameters, and metrics.

2.3.1 Notation

Let us consider a mobile ad hoc network represented by a graph $G(V, E)$ consisting of a set of nodes V and a set of edges E, where $V = \{node_1, node_2, \ldots, node_z\}$ and an edge (u, v) can connect $u, v \in V$, in some network of diameter, D. The diameter of MANET is the path with the smallest number of hops between the furthest two arbitrary nodes in the network [67]. An edge (u, v) is present in the network if and only if the transmission of u is heard by v successfully and

vice versa.

Let $s \in V$ be a source node and define a function, $h_s : V \to \mathbb{N}_0$ where $h_s(u)$ is the hop count between s and some other node $u \in V$ and $0 < h_s(u) \leq D$ and $h_s(s) = 0$. Let us assume that m and n are two positive integers where $0 \leq m < n \leq D$. The subset of all nodes, $u \in V$, for which $m < h_s(u) \leq n$ will be called a region with respect to s. A sequence of positive integers β_i where $0 < i \leq k$, $\exists\, \beta_k = D$ and $\beta_{i-1} < \beta_i\ \forall i, 1 < i \leq k$ defines a set of disjoint regions of V with respect to s. In general, the region τ_i is the subset of all nodes, $u \in V$, for which $\beta_{i-1} < h_s(u) \leq \beta_i$. The depth of τ_i is equal to $\beta_i - \beta_{i-1}, \forall i, 1 < i \leq k$.

Table 2-1: Table of nomenclature.

Parameter	Meaning
s	Source node
d	Destination
f	Finder of a needed route (intermediate node or destination)
D	Network diameter
$h_s(u)$	Hop count between *s* and node *u* in the same network.

2.3.2 Justification of the methods used

In real experiments the whole system is tested in real world settings and this needs a budget and manpower. So far, there has been a little work on the deployment and performance measurement of real world MANETs [82]. On the other hand, simulation and mathematical modelling play important roles in performance evaluation [42, 43] where real experimentation isn't feasible providing reasonable performance measures with a minimum amount of effort and cost [31]. For this reason, they were selected as the methods of study in this thesis.

Simulation and mathematical modelling are valuable tools for studying MANET systems. However, those tools always require certain assumptions for simplification (e.g. on radio properties and nodes mobility) in order to keep the simulation model's complexity at a manageable level. As a result, the model may not capture all the factors that might affect system performance.

Mathematical model

Mathematical models allow the network analyst to evaluate the network by deriving a set of equations that predicts the performance and gives insight into how different factors affect the performance of the network. However, when the system is up and running it is often difficult, time consuming, and expensive to make changes if performance problems are encountered. A detailed complete mathematical model for multi-hop networks with reasonable assumptions is coarse in nature [96]. Nevertheless, we have tried to model packet delay analytically in all our proposed algorithms after adopting certain simplifying assumptions (e.g. no mobility).

The average packet delay is one of the most important performance measures in network systems because delay considerations influence the choice of network algorithms such as routing [13]. Furthermore, it is very important to understand and analyse delays in any network before implementing the proposed algorithm. Networks can be modelled mathematically using a modelling tool such as queuing theory [13], Petri nets [14, 97], and finite state machines [16]. Queuing theory is a primary methodological framework to analyse network delay. It can be used as a mathematical modelling method to represent a MANET as a network of queuing systems [46, 57, 109]. Packets are subject to queuing delay when waiting to be processed and transmitted. So the whole system can be modelled as a network of queuing systems which operate in steady state.

When the network gets congested, the channel contention increases which in turn increases the system delay and incurs more packet loss [73] leading to a severe degradation in network performance. Understanding the relationship between congestion and delay in any network is essential especially in a resource-limited environment like MANETs. Thus modelling our system using queuing theory [13, 63] provides us with better understanding of the delay in our systems.

Simulation model

Simulation provides a way of predicting performance in the absence of a real network that can be used for performance measurement. It gives us insight and understanding of our algorithms' performance within the timeframe and budget. An accurate observation taken from a test-bed or real life implementation is potentially very costly and needs long time [19] with limitation in size. Also simulation has advantage in measuring performance over a real network implementation

because simulation can be repeated for different versions of the proposed algorithm under the same condition then compare their performance [43] at easy cost. Furthermore, simulation allows an analyst to evaluate performance under different network conditions and traffic loads [52, 99]. So simulations have been conducted to evaluate the new algorithms and some other existing algorithms that are related to our work in a comparative study.

Since nodes are mobile in MANETs, modelling these movements is not obvious. In order to simulate a new protocol, it is necessary to use a *mobility model* that reasonably represents the movements of a typical node [19]. Accurate mobility models should be chosen carefully to determine whether the proposed protocol will be useful when implemented or not. Moreover, one of the main characteristics of mobility in MANETs is the maximum speed of nodes because the speed of nodes determines the rate of broken links which increase the overhead in on-demand protocols.

Mobility models used in the simulation of MANETs are based on real trace or synthetic models [19, 67]. Trace-driven models are useful and accurate if they are obtained through long observation in the field for particular scenarios involving real user participants but are not always available because they are costly and time-consuming to accumulate. On the other hand, synthetic models do not provide such accuracy but in attempting to model realistic user mobility behaviour, they enable researchers to estimate behaviour in the absence of real trace models. In this thesis synthetic models of mobility are used. Synthetic models have been classified in [35] into entity and group mobility models depending on whether individual nodes or a group of nodes are concerned.

In MANETs, the entity mobility models typically represent nodes whose movements are completely independent of each other, e.g. the Random Way Point (RWP) model [61]. However, a group mobility model may be used to simulate a cooperative characteristic, such as working together to accomplish a common goal. Such a model reflects the behaviour of nodes in a group as the group moves together, e.g. Reference Point Group Mobility (RPGM) model [10, 12, 53].

In the RWP model, each node at the beginning of the simulation starts by being stationary for a pause time then chooses a random destination within the simulation arena and starts moving

towards the chosen spot with a random speed chosen from a uniform distribution [minimum speed, maximum speed]. After the node reaches its destination, it stops again for a pause time interval and chooses a new destination and speed. All nodes follow this pattern until the end of the simulation run. The RWP model takes time to reach a stable distribution of mobile nodes so the modified RWP model [86] is used in this thesis to take care of this node distribution problem.

In the RPGM model, group movements are based upon the movement of the group reference point following its direction and speed with speed selected randomly within the range [minimum speed, maximum speed]. At the start, each member of a group is uniformly distributed around their reference point (the group leader). Afterward, every node has a speed and direction derived and randomly deviating from that of their reference point by a *Speed Deviation Ratio* (*SDR*) and *Angle Deviation Ra*tio (*ADR*) where $0 \leq SDR, ADR \leq 1$. Moreover, nodes move randomly within their group where *SDR* and *ADR* are used to control the deviation of the velocity (speed and direction) of group members from their leader's velocity. Spatial locality between members of the same group can be obtained by using a very small value for both parameters *SDR* and *ADR* such as $SDR, ADR \leq 0.1$ [12].

2.3.3 Simulation Environment

Several discrete event network simulators have been developed, commercial or non-commercial, for performance analysis in MANETs. Commonly used network simulators include ns2 [41], GloMoSim [128], OMNET++ [118], CNET [76], and OPNET [28]. To conduct our simulation experiments, the Network Simulator (ns2) has been chosen as a simulation tool, which has been heavily used in research studies on MANETs [34, 50, 71, 73, 78, 79, 104, 132], because it includes detailed simulation of the important operations of ad hoc networks and well documented in the literature. It is flexible since it is open source free software. The algorithms were implemented using ns2 simulator version 2.29 [41]. The main modifications were done to files listed in Figure 2-1.When modifying the needed ns2 source code, special care was taken to ensure that the algorithms function correctly and that the simulator would not exhibit unwanted side effects; this was done through detailed use of the validation suite provided with ns2, before and after modifications, as well as gradual testing of the implemented features.

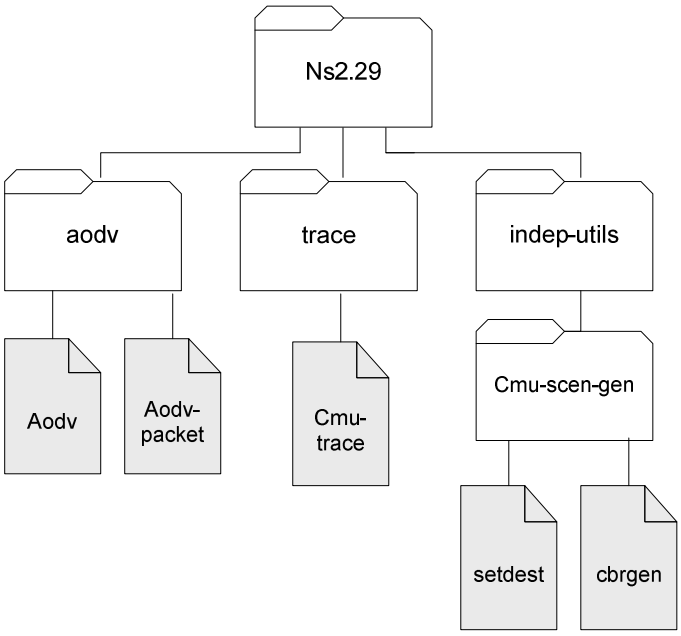

Figure 2-1. The modified files in ns2.29.

Although AODV has been used in all our simulation runs as the base on-demand routing protocol, the techniques implemented in this thesis are generic in nature thus applicable to other on-demand routing protocols regardless of broadcast mechanisms used, deterministic or probabilistic.

All nodes are assumed to be equipped with the same transceiver i.e. IEEE 802.11 where IEEE 802.11 standard operates at data rate up to 2Mbps. The IEEE 802.11 MAC layer provides two access methods to the wireless media: the Distributed Coordination Function (DCF) and the Point Coordination Function (PCF) [54] where the former is contention-based and the latter is contention-free. The DCF is the fundamental MAC access method that works in a distributed fashion which makes it suitable for MANETs that have neither infrastructure nor central management. PCF is an optional access method built on top of the DCF relying on a central node and hence is suitable for infrastructure wireless network. DCF is based on the Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) scheme. CSMA is a contention-based algorithm which ensures that each node senses the medium before sending, to avoid collisions and retransmissions. In addition to physical carrier sensing the DCF has a virtual carrier sensing phase that exchanges Request-To-Send/Clear-To-Send (RTS/CTS) [41] control packets as a handshaking mechanism between neighbouring nodes before transmitting unicast packets to reduce the probability of collisions due to hidden terminals problem [4].

This simulation model is represented by two scenario files, which are the mobility and traffic scenario. The topologies are generated by using different mobility scenarios because those scenarios correspond to how nodes are distributed over the simulation area and their movement during simulation time where nodes can move at any time without notice as this is a normal behaviour in MANETs. The traffic scenario files contain information such as: connection type, number of connections, packet size, and traffic rate.

The RPGM mobility generator [11] was used to generate mobility scenarios for all of our simulation runs since it models the random motion of groups of nodes and of individual nodes within the group. Mobility scenarios for reference points of all the groups, one reference point per a group, are generated using the RWP model; then these scenarios are fed to the RPGM mobility generator to generate the needed sets of scenarios for the other nodes using the value of 0.5 for both *SDR* and *ADR* to avoid spatial correlation within each group thus reducing clustering and network partitioning. Moreover, each group contains 10 nodes. The minimum speed is 1m/s and pause time is 50s to simulate a pedestrian taking short rest.

A traffic generator was used to simulate constant bit rate (CBR) with a packet data payload of 512 bytes. Data packets are transmitted at a rate of four packets per second. CBR/UDP was chosen as a communication service due to its simplicity and predictability that gives us a better chance to test our algorithms during the experiments where our main concern is the route discovery process. Moreover, communication sessions were injected to simulate traffic in a network that exhibit traffic locality where each five flows were between one source node and different destinations within a group of ten nodes to give the neighbourhood of the source node a chance to expand or shrink. The source node and the five destinations are randomly selected. All CBR connections were started at a random time during the first 180 seconds of simulated time where most of the route discoveries were initiated. All connections remain active through the entire simulation.

Nodes are assumed to operate in a flat outdoor area so the propagation model has been devised by experts in modulation thus we used both a free space propagation model and a two-ray ground reflection model. The Two-ray Ground model [40] was utilised as a radio propagation model in all of our simulations because it considers both the single line-of-sight path and a ground reflection path. When two-ray ground is used as radio propagation model in ns2, the system uses Friss-space

attenuation at near distance and two-ray ground at far distance depending on the distance between transmitter and receiver [41]. This model implements Omni Directional Antenna module which has unity gain for all direction.

Nodes in ad hoc network may run out of power or switch themselves off to save energy. However in the simulated scenarios nodes are assumed to have sufficient power to fully operate throughout the simulation time to allow us to study the behaviour of the new algorithms under the same environments and allow direct and fair comparisons between the new algorithms and the existing without losing nodes, however it would be interesting to study the energy consumption as a next step of this research.

Also, we assume that links are bidirectional where the finder of the needed route updates the destination by sending a unicast gratuitous route reply; nodes are willing to cooperate in the routing protocol as relay nodes. Furthermore, dealing with security threats such as malicious attacks or denial of services is important for the operation of any network [84, 117]. However, we assume that dealing with security attacks is done with the help of a security protocol [9, 85].

2.3.4 System Parameters

As in the previous studies of [6, 20, 30, 102, 103], the simulation model consists of the following main components: simulation area, simulation time, number of nodes, mobility model, maximum node speed, and number of traffic sessions. All nodes are identical, mobile, and assumed to operate in a squared simulation area of 1000m x1000m. The transmission range is fixed to 100m in all nodes to approximately simulate networks with a minimum hop count of 10 hops between two border nodes on opposite sides in a connected network. Each run was simulated for 900 seconds of simulation time to avoid immature termination and keep the simulation time manageable, ignoring the first 30 seconds as a start-up period for the whole network to analyse it in steady state and avoid counting all start-up control packets such as ARP packets. For each topology, 30 runs were performed then averaged to produce the graphs shown throughout this thesis and a 95% confidence interval is shown as standard error bars in the relevant figures. Error bars are shown in some figures but not all for the sake of clarity of presentation. Table 2-2 provides a summary of the chosen simulation parameter values. These setting could represent MANET scenarios in real

life such as groups of tourists visiting a historical site following their tour guides. Although the number of tourists in one group could be larger than the one presented in these scenarios and the operational time could be longer but this is to keep the simulation manageable in terms of time.

Table 2-2: Summary of simulation parameters.

Parameter	Value
Transmission range	100m
Topology size	1000m x 1000m
Simulation time	900s
Packet size	512 bytes
Packet rate	4pkt/s
Traffic load	5,10…35 sessions
Traffic type	CBR(UDP)
Routing protocol	AODV
Number of Nodes	20,30,..,100
Number of runs per point	30
Antenna type	Omni Antenna
MAC protocol	IEEE 802.11with RTS/CTS
Maximum speed	2,5,7,10,13,15m/s
Minimum speed	1m/s
Pause time	50s
Mobility model	RPGM model
SDR	0.5
ADR	0.5
Propagation model	Two-Ray Ground model

In our simulation, we concentrate on three major parameters: network size, traffic load, and maximum speed in three different cases by varying one parameter while keeping the other two constant as explained below and summarised in Table 2-3:

- *Network Size*: is used to study the effect of varying network size on network performance. Network size is the total number of nodes in the network. When the network size increases, the average hop count of routes also increases which may increase network latency and routing overhead. The simulation area is kept constant in all scenarios to study our algorithms' performance in both small and moderate size environments, since we are interested in knowing their behaviour in both kinds of environments where

moderate size networks reduces the chance of network partitioning. Simulation has been performed using nine topologies with different number of nodes, multiples of 10, from 20 to 100 while fixing the traffic load to ten communication sessions and the maximum speed to 15m/s.

- A network of size 20 nodes is used as small size network.
- A network of 100 nodes is used as moderate size network.

- *Traffic Load*: is used to study the effect of varying the amount of traffic load. The traffic load of sizes 5, 10, 15… 35 communication (data) sessions were used in some simulations with a size network of 70 nodes to avoid sparse and dense environments and maximum speed of 15m/s. The purpose is to test our algorithms using reasonably incremented amount of traffic while avoiding saturation and keeping the simulation at a manageable level. We managed to run up to 50 communication sessions. Runs with communication sessions greater than 35 did not show any changes in overall performance but need a considerable a mount of time to run. The number of route requests broadcast increases with more traffic load which increases latency, congestion, and packet loss.
  - Traffic load of 5 data session is used in *light traffic* network
  - Traffic load of 35 data session is used in *heavy traffic* network.
- *Mobility*: is used to study the effect of varying the maximum speed where mobility affects network connectivity which has an impact on the network performance. The maximum speeds used are 2, 5, 7, 10, 13, and 15m/s to simulate human speed as well as vehicle movement with network of size of 70 nodes and traffic load of 10 communication sessions.
  - A slow speed network has a maximum speed of 2m/s.
  - A fast speed network has a maximum speed of 15m/s.

Table 2-3: Simulation parameters for the three cases used.

Cases	Simulation parameters		
	Network Size	Traffic Load	Maximum Speed
Network size	20, 30… 100 nodes	10 sessions	15m/s
Traffic load	70 nodes	5, 10, 15… 35 sessions	15m/s
Mobility	70 nodes	10 sessions	2, 5, 7, 10, 13, 15m/s

2.3.5 Performance metrics

The performance of the route discovery process can be measured by studying latency, overhead, and congestion. In this thesis, the terms "delay" and "latency" will be used interchangeably. The latency can be studied by analysing the end-to-end packet delay and the average of route request latency per hop while the overhead can be measured by studying the routing overhead and the congestion level can be determined by analysing packet loss.

Latency:

- End-to-end delay (ms): the application data can experience queuing delay in the source node until the needed route is discovered so the end-to-end delay is the route discovery time plus all delays that the data experience from the time it was sent by a source node until the time it was received at the destination. Moreover, route discovery time is the round trip time of route request and route reply between source node and finder of the needed route.

- Route request latency (ms): the average of delays per hop among all route requests in a single simulation scenario. Latency of one route request is the average delay experienced by the route request per hop from the time it was sent by a source node until the time it was discarded.

Overhead:

- Routing overhead (packets): the *route request overhead* plus the number of chase packets received in the whole network. The route request overhead is the number of route request packets received in the whole network, where every reception of a route request or a chase packet at any hop contributes one to the total. If the algorithm does not use chase packets then routing overhead = route request overhead.

Congestion:

- Packet loss (packets): the number of dropped packets in the whole network. In MANETs, congestion and mobility are the main causes of packet loss [73].

2.4 Summary

While Chapter 1 has provided the context and the motivation behind undertaking this research work, this chapter completed the presentation of the background information and related work necessary for a clear understanding. The AODV algorithm was explained as an example of an on-demand routing protocol, along with a general overview of the existing broadcasting algorithms proposed in the literature for MANETs. Also, simple flooding and the broadcast storm problem were explained and route discovery improvement techniques that avoid full network coverage were discussed.

This chapter has provided the preliminaries required throughout the thesis, including: notation, justification of the methods for the performance analysis, explanation of the simulation environment, assumptions, parameters, and performance metrics.

Chapter 3: Traffic Locality in MANETs

3.1 Introduction

MANETs are very useful in applications that need immediate collaboration and communication with the absence of network infrastructure where a temporary connection can be established for quick communication [56, 84]. Such collaborative jobs often demand traffic to be between known source-destination pairs to accomplish specific tasks. So if this pattern of traffic is found in an application then the design of the algorithm should utilise it.

The principle of locality was first applied in memory referencing behaviour [37] then it was subsequently observed in the use of other resources such as file referencing [111]. The locality of reference concept deals with the process of accessing a single resource more than once at points in some sense "close" to each other in either time or space. It includes *spatial* and *temporal* locality [65].

- *Spatial locality:* a resource has a higher chance of being referenced if a neighbouring resource was just referenced.
- *Temporal locality:* a resource that is being referenced now will be referenced again sometime in the immediate near future.

In memory management, locality of reference is the principle behind caching, where some instructions and data are placed in higher-speed memory to exploit the probability that future accesses will exhibit locality of reference [36]. In networking, locality is observed through the fact that devices, e.g. hosts or routers, within the same geographical area tend to communicate for a while more often than those that are further apart, and exhibit both temporal and spatial locality [112].

The importance of traffic locality concept is recognised in networking. Traffic locality concept is a motivation factor behind network clusters and workgroups [17, 58, 83]. It is used in local area networks [47] where it was defined as the distribution of packet references over time and space.

BitTorrent protocols are used to improve traffic locality in server-client architecture [15]. It is a peer-to-peer file sharing (P2P) communications protocol used for distributing large amounts of data widely without the original distributor incurring the entire costs. It reduces the cost and burden on any given individual source, provides redundancy against system problems, and reduces dependence on the original distributor. This is achieved by making use of the upload bandwidth of all nodes (called peers) downloading the file.

In infrastructure wireless networks, traffic locality is utilised to improve load balancing in base stations [84, 98] where it is defined to be the amount of terminated traffic within one cell. When a clustering algorithm is used to impose a hierarchical structure in a MANET system, the locality of traffic within the same cluster is the key factor on deciding the feasibility of large ad hoc networks [58] because is node communicates mostly with other nodes within the same cluster in the present of spatial locality [83].

3.1.1 Locality in MANETs

In MANETs, locality is likely to be observed through the fact that neighbours, nodes in the same geographical area, tend to receive communication from the same sources, highlighting the spatial locality [20, 107]. Also, nodes communicated with in the near past have high probability of re-communicating with in the near future leading to temporal locality [20, 98]. Certain MANET's applications may exhibit traffic behaviour that we called traffic locality. It follows particular patterns in which the source node tends to communicate with certain set of nodes more than others; regardless of their locations or time of communication. The traffic locality might include either spatial or temporal or both. This observation has motivated us to introduce a new form of traffic locality in MANETs, which is the subject of this chapter, where the traffic locality of a particular source node is captured in its working set which is simply the set of nodes that the source node is mostly communicating with, not necessarily neighbours. Members of the working set may change over time.

Traffic locality, based on the concept of "working set", identifies the set of nodes that a given source is mostly communicating with. These nodes are not necessarily identified by space or time but rather by intensity of traffic within the working set over some time interval. So, this set

performs the working set for that source node at a particular instance of time because the set members change with time according to the communication needs. Moreover, if a source exhibits traffic locality with a certain destination, the intermediate node comprising the route in question will also be a member of the source node's working set until one of them moves far away.

Applications in MANETs exhibit traffic locality due to the communication requirements of the users carrying and operating the nodes. One common application that exhibits traffic locality in MANETs is group communication ad hoc network [81] where a group of nodes communicates with each other to accomplish a common goal.

3.2 Traffic Locality oriented Route Discovery approach (TLRDA)

In this thesis, traffic locality concept is utilised to improve the route discovery process in on-demand routing protocols for MANETs running applications that exhibit traffic locality. This concept is used to develop a new approach that we called traffic locality oriented route discovery approach, TLRDA, to improve the route discovery process in on-demand routing protocols. It works by gradually building up the node neighbourhood as a region centred at the source node and expected to contain most of the members of its working set where the whole connected network consists of two disjoint regions: τ_1 represents *neighbourhood*, and τ_2 represents *beyond-neighbourhood* from each source node prospective. Since the neighbourhood region contains the source node's working set, no extra delays are added in this region to avoid delaying the route discovery process. On the other hand, delaying a fulfilled route request in the beyond-neighbourhood region adds no latency to the discovery process.

Establishing such neighbourhood is a challenging endeavour as it adapts to the traffic in an effort to build then maintains the neighbourhood region reflecting the current working set. Upon joining the network, the neighbourhood region for this new node will not be established so the new node needs a start-up period during which it uses the original broadcast algorithm depending on the routing algorithm used. Once the neighbourhood region is reasonably estimated, an intermediate node broadcasts any route request generated from that source node to all nodes within the source node's neighbourhood region without adding any delay in an effort to minimise the route

discovery time.

Due to the scarce resources in MANETs, the approach is kept simple by avoiding the collection or manipulation of large amount of data. Also, the global information is avoided because it is unavailable in a real environment that uses no external resources. Thus, each source node has a locality parameter *LP* where $LP \in \mathbb{N}^*$ which corresponds to the current estimated depth of its neighbourhood which might be defined by the *weighted average* of hop counts between that source node and destinations. A node, *x,* is considered to be part of the neighbourhood set of a source node, *s*, if $h_s(x) \leq LP$.

The query of a needed route can be answered by the destination or any intermediate node in the way to destination. The term route finder, *f, refers* to the first node that finds the route in its cache table whether it is the destination or an intermediate node. Figure 3-1 illustrates an example when the route finder is an intermediate node.

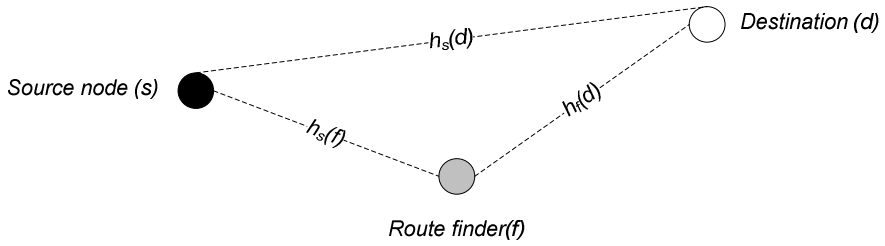

Figure 3-1: The finder of a requested route between source and destination.

Formally considering region-partition, the two regions $\{\tau_1, \tau_2\}$ are the neighbourhood and beyond-neighbourhood respectively in a network running applications that exhibit traffic locality. It is obvious that the two regions are disjoint sets so $\tau_1 \cap \tau_2 = \emptyset$. Let us consider a source node *s*, any node $v \in \tau_1$ satisfies the condition $h_s(v) \leq LP_r$ and any node $u \in \tau_2$ should satisfy the condition $h_s(u) > LP_r$ from that source node prospective where LP_r is the locality parameter for the source node s. *LP* is continuously tuned to adapt to the current situation using the values of $h_s(d)$ for all needed routes with respect to *s*, whether a route is discovered by an intermediate node or the destination itself.

The approach is adaptive and each active source node adjusts its locality parameter, *LP*, to expand or shrink the neighbourhood boundary. If a route finder is outside the neighbourhood then this

requires the neighbourhood to be adjusted via some adaptation strategy. One possible strategy is as follows: *LP* is adjusted by taking the weighted average of the current value of *LP* and the new hop count from the last successful route discovery where the initial value of *LP* is 1. Alternatively, any monotonically increasing or decreasing function could be used but this lacks a countervailing expanding or shrinking ability respectively so will not be considered. Figure 3-2 shows instances of the source node *s* neighbourhood shrinking and expansion abilities. In analogy, the expansion and shrinking correspond to swabbing pages in and out in the context of memory management.

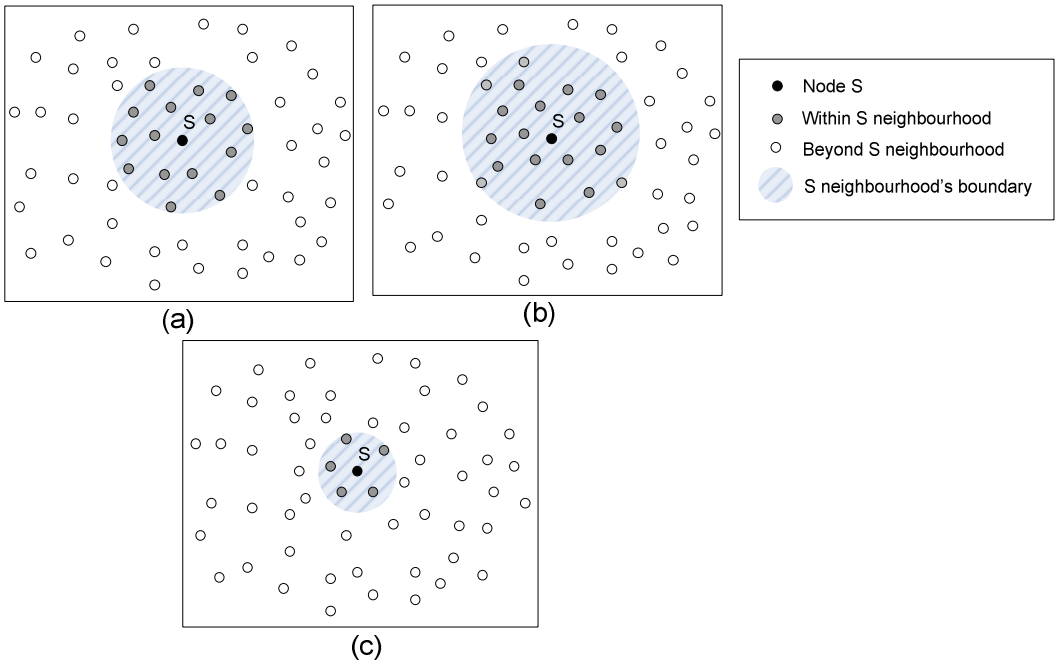

Figure 3-2: Neighbourhood expansion and shrinking. (a) Neighbourhood when *LP* = 2 hops. (b) Expanding when *LP* = 3 hops. (c) Shrinking when *LP* = 1 hop.

To calculate *LP*, the source node needs to store the number of its previous route requests. To illustrate the neighbourhood adjustment process, let us consider the source node *s* at any time after completing its start-up phase. When s receives a reply answering its current query it updates its *LP* using equation (3.1) after extracting $h_s(d)$ from the received route reply packet where y is the number of previous route requests that already been sent by s. If $h_s(d) \geq LP_{old}$ then the neighbourhood of *s* expands; otherwise it shrinks.

$$LP_{old} = \alpha \times LP_{old} + (1-\alpha) \times h_s(d) \qquad where\ \alpha = \frac{y}{(y+1)}$$

$$LP_{new} = \begin{cases} \lceil LP_{old} \rceil & h_s(d) \geq LP_{old} \\ \lfloor LP_{old} \rfloor & h_s(d) < LP_{old} \end{cases} \tag{3.1}$$

In TLRDA, a source node broadcasts a route request after adding the value of its *LP* to the route request packet so intermediate nodes can decide if the route request is within its source node's neighbourhood or not. To avoid ambiguity we will use LP_r to refer to the *LP* value stored in the route request. Figure 3-3 shows the outline of the procedure used to update the locality parameter *LP* at the source node in TLRDA. The steps for updating the locality parameter *LP* are stated in lines 5 to 9 where those steps are performed after receiving the route reply to prepare the source node for the next route request. The function Ceiling (line 9) returns the smallest integer greater than or equal to its parameter while the function Floor (line 7) returns the greatest integer less than or equal to its parameter. To avoid stale information and prevent α from approaching 1 as *y* gets large due to $\lim_{y\to\infty} \frac{y}{y+1} = 1$, where only the function Ceiling or Floor affects the value of *LP*, each active source node needs to reset its local parameter *y* (line 2) to its initial value, *Initial_y*, when *y* reaches its maximum value, *max_y* (as in line 1). Each time y is initialised to 1, the partial historical information represented by LP_{old} is given the same weight as the hop count. Alternatively, if *y* is initialised to zero then the value of α will be zero leading to full weight of 1 to the hop count.

Steps preformed by source node upon receiving a route reply
y = previous number of route requests.

```
1:  If y >= max_y then
2:  y = Initial_y
3:  End if
4:  α = y/(y+1)
5:  LP_new = αLP_old + (1 - α)h_s(d)
6:  If h_s(d) < LP_old then
7:      LP_new = Floor(LP_new)
8:  Else
9:      LP_new = Ceiling(LP_new)
10: End if
11: LP_old = LP_new
12: y =y+1
```

Figure 3-3: Outline of the Update procedure for the locality parameter LP at the source node in TLRDA.

Example:

To illustrate the idea of updating *LP*, let us assume that the current value of *LP* is 4 where the

source node is ready to send the fifth route request thus $\alpha = \frac{4}{5}$. *Initial-y* = 1 and *max-y* = 10 assuming that the average discovery time is 1 unit of time. After sending the 6th route request assuming the route finder for such request is 4 hops from the source, the value of *LP* will stay unchanged. However, when the next routes finders are at 6, 7, and 8 hops from the source; the neighbourhood will be expanded to 5, 6, and then 7 respectively. This happens because the neighbourhood is expanded to accommodate more of the routes finders and destinations. Furthermore, the algorithm will continue to adjust according to the new values of $h_s(d)$ by expanding with the growth of $h_s(d)$ and shrinking with the decline of $h_s(d)$ whenever is needed as shown in Table 3-1.

Table 3-1: The updating of LP starting at LP = 4.

Previous Route Request (y)	Hop Count $h_s(d)$	LP_{new}
5	4	4
6	6	5
7	7	6
8	8	7
9	7	7
10	8	8
1	6	7
2	9	8
3	9	9
4	6	8

3.3 Conclusions

In MANETs, certain applications may exhibit traffic behaviour that we called traffic locality following particular patterns in which the source node tends to communicate with certain set of nodes more than others regardless of their locations or time of communication and it might include either spatial or temporal or both. The traffic locality of a particular source node ties with its working set.

TLRDA works by establishing a neighbourhood that includes the most likely destinations for a particular source node. The source node broadcasts the route request without adding any delay within its neighbourhood region. In an effort to improve the route discovery process in applications that exhibit traffic locality for MANETs, the new adaptive route discovery algorithm gradually builds up the node neighbourhood as a region, with the ability to change, centred at the source node and expected to contain most of the members of its working set.

The concept of traffic locality that was established in this chapter will be used as the base for developing three new route discovery algorithms namely: TLRDA-D, TLRDA-C, and TL-ERS will be described in Chapters 4, 5, and 6 respectively.

Chapter 4: Traffic Locality oriented Route Discovery Algorithm with Delay

4.1 Introduction

The route discovery algorithms which are described in this research are aimed for MANET applications that exhibit traffic locality. The approach TLRDA that was introduced in Chapter 3 earlier, establishes neighbourhood and beyond-neighbourhood regions for each active source node. The neighbourhood region includes most of the likely destinations for a given source node. Route request is broadcast without adding any delay within its source node's neighbourhood boundary to discover routes quickly.

In this chapter, a new algorithm is proposed. It aims at reducing the end-to-end delay by lowering the channel contention. It works by broadcasting the route request at different speeds depending on the region within which the route request is travelling; with respect to its source node. Moreover, the route request resides in the network until it fades away when reaching a boundary node or discarded when the time to live (TTL) field reaches zero.

4.2 The Traffic Locality oriented Route Discovery Algorithm with Delay (TLRDA-D)

A new traffic locality oriented route discovery algorithm with delay, TLRDA-D for short, is proposed where D stands for a delay e.g. TLRDA-k denotes an instance of the algorithm where the delay (D) equals to k units of time. TLRDA-D is based on TLRDA approach, introduced earlier, that utilises the concept of traffic locality to establish a neighbourhood. Intermediate nodes in TLRDA-D broadcast route requests without adding any extra delay while route request packets are propagating within the neighbourhood boundary. However beyond this boundary, nodes broadcast route requests with a delay at each node until the route request broadcast fades or its TTL reaches zero.

The motivation for adding this delay within the beyond-neighbourhood region is to give a higher

priority to route requests that are travelling within their own source node's neighbourhood regions. Moreover, other route requests that are travelling within their source node's beyond-neighbourhood regions have a higher chance of being already fulfilled and thus given lower priority. As will be shown below, this approach not only improves the average route discovery time but also improves the latency of the whole network, as it generates less contention throughout the network.

The delay will be calculated by a monotonic increasing function of LP as the route request propagates further within the beyond-neighbourhood region, since the chance of route request fulfilment increases with each hop when the route request moves away from the source node's neighbourhood region. Five instances of TLRDA-D have been considered in this chapter with different amounts of delay (d_x) where d_i stands for the i^{th} instance. Moreover, the delay increment can be logarithmic, linear, polynomial, or exponential. However, the exponential increase yields a huge amount of delay which makes it unsuitable for resource-sensitive environment like MANETs and hence it is ruled out. Simulation is used to help us decide on the relative effectiveness of other possible increment functions for the delay added to the route request dissemination in the beyond-neighbourhood region for TLRDA-D and whether it should be logarithmic (d_0), linear (d_1, d_2, and d_3), or polynomial (d_4). TLRDA-D instances have been implemented using five different amounts of delay as stated in Table 4-1 where d_i at any intermediate node takes the following values:

$$d_i = \begin{cases} \log_2(LP) & i = 0 \\ 2^{i-1}LP & i = 1, 2, 3 \\ LP^2 & i = 4 \end{cases} \tag{4.1}$$

If a route reply is not received within an estimated period of time called NETwork Traversal Time (*NETTT*), the source node tries again to discover the route by broadcasting another route request for a pre-specified maximum number of tries depending on the on-demand routing algorithm used. So the source node waits *NETTT* units of time to receive a reply before trying to search for the destination again. Assuming the worst-case scenario where Node Traversal Time (*NTT*) follows the on-demand routing algorithm used, TLRDA-D calculates this estimated time as follows:

$$NETTT = 2\{(LP * NTT) + (D - LP)(NTT + d_i)\} \tag{4.2}$$

Table 4-1: The amounts of delay imposed in all five instances of TLRDA-D.

Algorithm	Amount of delay
TLRDA-d_0	$d_0 = log_2(LP_r)$
TLRDA-d_1	$d_1 = LP_r$
TLRDA-d_2	$d_2 = 2LP_r$
TLRDA-d_3	$d_3 = 4LP_r$
TLRDA-d_4	$d_4 = {LP_r}^2$

Upon receiving a route request for the first time in on-demand routing algorithms, the intermediate node stores the broadcast ID plus the source node IP address in a table for an estimated time which we called Broadcast Cache Time (*BCT*) as a part of the route request processing steps. The broadcast ID and the source node IP address, extracted from the route request packet, uniquely identify a particular route request so this information is used to distinguish between new and redundant route requests. When BCT expires, the route request record is deleted from the table. BCT is calculated in TLRDA-D as follows at the intermediate node m:

$$BCT = \begin{cases} BCT & h_s(m) \leq LP_r \\ BCT + d_i & h_s(m) > LP_r \end{cases} \quad (4.3)$$

Upon receiving a route request, each node performs the steps shown in Figure 4-1. If the route request has been received before, then it is considered redundant thus discarded. Otherwise, the receiving node compares *LP* value from the route request packet with the hop count after counting itself as an extra hop. If the node resides in the beyond-neighbourhood region of the route request initiator, the node holds the route request for d_i units of time then processes it. Otherwise, the node processes the route request according to the routing algorithm used.

Steps preformed by each node upon receiving a route request in TLRDA-D

```
1:  If i = 0 then d = log2(LPr)
3:  Else if i = 1 then d = LPr
4:      Else if i = 2 then d = 2*LPr
5:          Else if i = 3 then d = 4*LPr
6:              Else if i = 4 then d = LPr * LPr end if
7:              End if
8:          End if
9:      End if
10: End if
11: If route request is a duplicate
12:     Discard the route request
13: Else
14:     If hop_count > LPr then
15:         Wait d units of time
16:     End if
17:     Process the route request
18: End if
```

Figure 4-1: Processing of route request packets at each node in TLRDA-D.

4.3 Delay analysis

All packets (data or control) are subject to different amounts of delay while travelling from source to destination in any network such as queuing delay, processing delay, propagation delay...etc. These delays depend on many factors such as: energy level, packet length, and contention level at that particular time. Propagation delay between two adjacent nodes is assumed to be negligible in this analysis since packets in wireless communications travel at the speed of light where propagation delay = $\frac{\text{distance}}{(3*10^8)}$. However, other delays affect the network performance.

In MANETs, most of the delays experienced by a packet are in the medium access control (MAC) layer due to contention. The MAC protocol does not distinguish between data and control packets because there is no distinction used when using DCF in IEEE 802.11 standards [108]. There is one queue in the MAC layer where all packets (data or control) are queued and process as FCFS (FIFO). Packets may have different service times because they differ in size. So, the MAC layer protocol has no knowledge about the importance of the data coming from higher layer such as

control packets which are treated as normal payloads.

MANET can be modelled mathematically as a network of queuing systems [46, 57, 109] since a mobile node receives different kind of packets (data or control packet) with different lengths, queues them if needed, and processes them then transmits them. The whole system can be modelled as a network of queuing systems operating in steady state. These models provide the adequate base for delay approximation. However, queuing theory requires some assumption to simplify the case because analyses with real assumption might be extremely difficult [13]. For the sake of simplicity two assumptions were made:

- Packet generation and arrival at each node assumed to be independent and identically distributed (i.i.d).
- Each node has infinite buffers to avoid dropped packets.

Each node is modelled as M/G/1 system [13, 42, 63] that satisfies the following conditions:

- Service delays are independent and have a general distribution.
- Packets arrive at each node according to a Poisson process with the rate λ and independent of service time.

The system has a single server that serves packets in their order of arrival (FCFS). When the packet is ready to be transmitted, the node senses the shared physical media before attempting to transmit by performing the CSMA/CA access protocol at the MAC layer. This contention time is included in the service time. Nodes in TLRDA-D are analysed as M/G/1 systems with different arriving customers, packets, such as data or control packets.

In this thesis, delay analysis is conducted for route requests which are divided into two classes: *Class 1* and *Class 2* containing route requests propagating in their source node's neighbourhood and *beyond-neighbourhood* region respectively.

Route requests travelling in the beyond-neighbourhood region are stored for *d* units of time before joining Class 1 queue where they are treated as Class 1 packets. To simplify the analysis, let us

assume that separate buffers are maintained for Class 2 before joining the queue of Class 1. When the server is free and Class 1 is nonempty, the first packet in Class 1 queue enters the service. Figure 4-2 shows a representation of a node as a queuing system running TLRDA-D on top of the on-demand routing algorithm used.

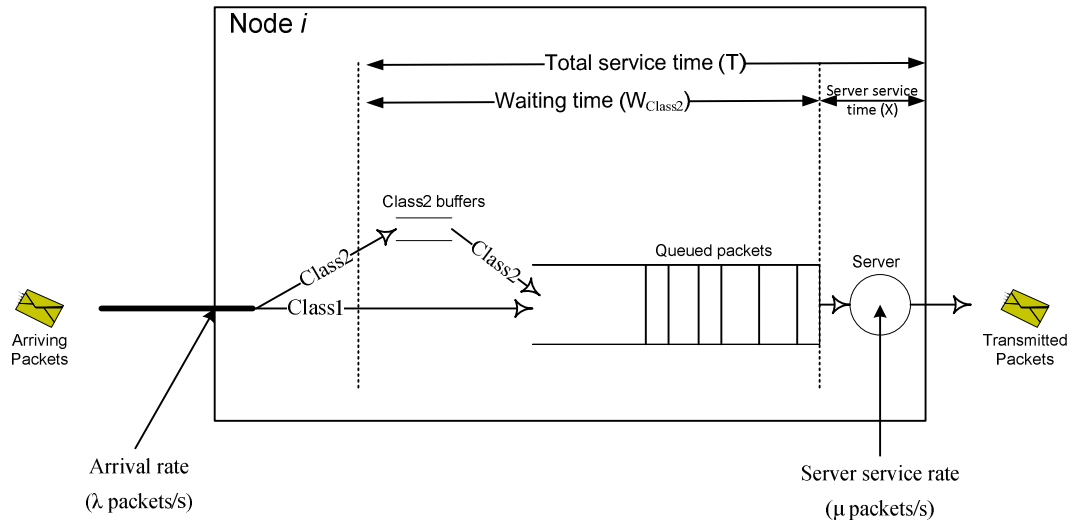

Figure 4-2: A mobile node in MANETs represented as a queue for TLRDA-D algorithm.

According to TLRDA-D, when a route request propagates in the beyond-neighbourhood region; an extra amount of delay should be added to give other packets a better chance of being transmitted earlier and to reduce the contention within other neighbourhood regions. Delaying route request packets when propagating in the beyond-neighbourhood region should not affect the discovery time of this route since most of the destinations of the route request lay within the neighbourhood region. In fact, fulfilled route requests compete with other packets to win the channel adding undesirable contention and should be given lower priorities over other data or control packets. To calculate the average waiting time for M/G/1 queue, a simple method from [13] is used. The notations used to perform the delay analysis for TLRDA-D are explained in Table 4-2.

In MANETs, packets are jittered by a random duration called Random Rebroadcast Delay [18] before being broadcast at each node. Random Rebroadcast Delay (RRD) is used to solve the broadcast storm problem by preventing broadcast synchronization where neighbouring nodes may transmit at the same time. RRD is generated uniformly in an interval between 1 and MAX-JITTER where MAX-JITTER is the maximum random number for RRD, e.g. MAX-JITTER in AODV is 10ms. The extra amount of delay imposed on route requests in the beyond-neighbourhood is

independent of RRD. RRD is used with broadcast packets whereas unicast packets face another kind of delay due to the handshaking mechanism. For simplicity we assume that these two delays are equal.

Table 4-2: Parameters of the queuing network model for TLRDA-D.

N	Average number of packets in the system
N_Q	Average number of packets waiting in the queue
N_i	number of packets waiting in the queue when the i^{th} packet arrives
λ	Arrival rate
μ	Service rate
ρ	Utilisation factor of the server $(\rho < 1)$
W_{Class1}	Average waiting time in the queue for Class 1 packet
W_{Class2}	Average waiting time in the queue for Class 2 packet
T_{Class1}	Average of total service time per packet for Class 1
T_{Class2}	Average of total service time per packet for Class 2
w_i	Waiting time for the i^{th} packet in the queue
R	Average residual service time
r_i	Residual service time is the remaining time of the packet currently in service when the i^{th} packet arrived
X	Average server service time
x_i	Server service time for the i^{th} packet
M	Average amount of jitter added to any broadcast packet
m_i	jitter added to the i^{th} broadcast packet
d	Amount of delay added to Class 2 packets
C	Channel contention

To derive the total service time for both Class 1 and Class 2 packets, let us consider the following:

Class 1 packet:

Class 1 contains route requests propagating in their source node's neighbourhood region. TLRDA-D processes Class 1 packets according to the routing algorithm used.

The service time for any packet $(x_1, x_2, \cdots)$ is a discrete random variable (r.v.) where the average service time $\overline{X} = \frac{1}{\mu}$

$$E\{X\} = \overline{X}$$

The average waiting time in the queue for the i^{th} route request (w_i) is consisting of service times (x_j) of all the packets currently waiting in the queue, residual time (r_i), and RRD (m_i):

$$w_i = \sum_{j=i-N_i}^{i-1} x_j + r_i + m_i \tag{4.4}$$

Since *M* is discrete r.v., the *k* moment, $\overline{M^k}$, of the jitter time is computed as

$$E\{M^k\} = \sum_{m} P(m)\, m^k$$

$$E\{w_i\} = E\left\{ \sum_{j=i-N_i}^{i-1} E\left\{x_j \middle| N_i\right\} \right\} + E\{r_i\} + E\{m_i\} \tag{4.5}$$

Knowing that N_i is a r.v. and independent of x_j.

$$E\{w_i\} = \bar{X} E\{N_i\} + E\{r_i\} + E\{m_i\} \tag{4.6}$$

Following the analysis in [13], all long-term averages viewed as limits when packet index converges to infinity assuming these limits exist. This assumption is true if $\rho < 1$. In other words, the arrival rate $(\lambda) <$ the service rate (μ) so the node can handle the packet received in reasonable time and avoid the unpleasant effect of saturation [63].

$$\lim_{i\to\infty} E\{w_i\} = \bar{X} \lim_{i\to\infty} E\{N_i\} + \lim_{i\to\infty} E\{r_i\} + \lim_{i\to\infty} E\{m_i\} \tag{4.7}$$

$$W_{Class1} = \bar{X} N_Q + R + M \tag{4.8}$$

Applying Little's Theorem as in [13]

$$N_Q = \lambda W_{Class1} \tag{4.9}$$

Substituting equation (4.9) in (4.8) and using $\rho = \bar{X}\lambda$:

$$W_{Class1} = \rho W_{Class1} + R + M \tag{4.10}$$

$$W_{Class1} = \frac{R + M}{(1 - \rho)} \tag{4.11}$$

Where the average residual time as stated in [13] is:

$$R = \frac{\lambda\ \overline{x^2}}{2} \tag{4.12}$$

The second moment ($\overline{x^2}$) of service time is computed as in [106]:

$$E\{X^2\} = \sum_{x_i} P(x_i)\, {x_i}^2$$

The average of waiting time formula can be obtained similar to [13, 63] by substituting (4.12) into (4.11):

$$W_{Class1} = \frac{\lambda\overline{x^2} + 2M}{2\ (1-\rho)} \tag{4.13}$$

Total service time for a Class 1 packet can be obtained from adding the waiting time in queue to the average server service time and the waiting time for the line to be free.

$$T_{Class1} = W_{Class1}\ + X \tag{4.14}$$

Class 2 packets:

When route requests travel in the beyond-neighbourhood region, they are delayed for *d* units of time at each node in this region. The average waiting time in the queue for the i^{th} route request is:

$$w_i = \sum_{j=i-N_i}^{i-1} x_j + r_i + m_i + d \tag{4.15}$$

Following the same analysis from equation (4.4) to (4.10):

$$W_{Class2} = \rho W_{Class2} + R + M + d \tag{4.16}$$

And by substituting R from equation (4.12) in (4.16):

$$W_{Class2} = \frac{\lambda\overline{x^2} + 2M + 2d}{2(1-\rho)} \tag{4.17}$$

In average, route requests that belong to Class 2 will experience a delay equal to the delay of other packets from Class 1 plus an extra amount of delay.

$$W_{Class2} = W_{Class1} + \frac{d}{(1-\rho)} \tag{4.18}$$

The total service time per packet for Class 2, T_{Class2}, is the average waiting time in the queue, W_{Class2}, plus the average server service time, X, which includes the waiting time for the channel to be free.

$$T_{Class2} = W_{Class2} + X \tag{4.19}$$

The average waiting time of T_{Class2} is more than T_{Class1} by $\frac{d}{(1-\rho)}$ units of time. This increment may not increase the end-to-end delay of the network due to the fact that the delay is added when the route request is outside the neighbourhood region. On the other hand, this delay reduces network congestion so the average service time, X, and the average waiting times W_{Class1} and W_{Class2} are reduced in both T_{Class1} as well as T_{Class2} compared to high congested network which indeed improve the network performance. The total contention TC is reduced when adding more delay to route requests propagation due to the reduction in congestion level. However, TC cannot be eliminated since it is attributed to other factors, beside congestion, such as fading and transmission errors. The contentions that are due to congestion and other factors are denoted by C and $\acute{C}$ respectively. Moreover, the total contention is computed as $TC = C + \acute{C}$. The channel contention ranges between some values C_{min} and C_{max} where $C_{min} \leq C \leq C_{max}$. So for a heavy congested network, such as networks using routing algorithms that use simple flooding, $TC = C_{max} + \acute{C}$ whereas in a well-controlled network $TC = \acute{C}$ since $C_{min} = 0$.

4.4 Mathematical formulation

MANETs consist of V nodes and each node process different kind of packets where T_{Class1} and T_{Class2} represent packet's delays within each node. For simplicity, the role of mobility is ignored in this delay analysis. Analyses of the route request latency and end-to-end delay are done for the whole network where stations of the queuing network corresponded to the nodes in MANET as in Figure 4-3.

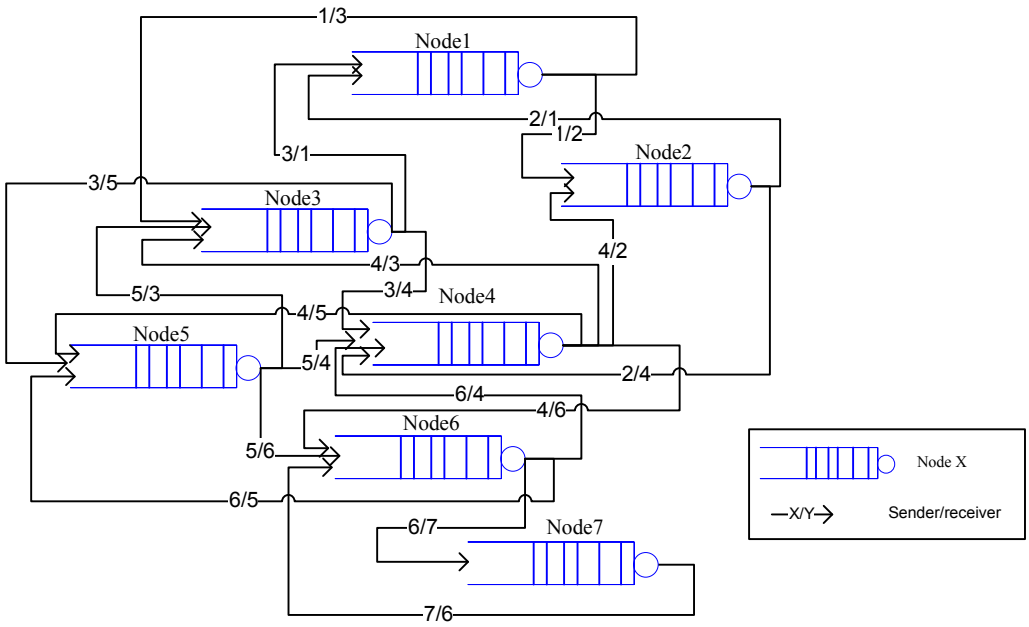

Figure 4-3: A mobile ad hoc network of size seven represented as a network of queuing systems.

A. End-to-end delay:

The end-to-end delay is the route discovery time plus the average delay experienced by the data packet from the time it is sent by the source node until it is received at the destination. Route discovery time is the round trip time of route request and route reply between source node and finder of the needed route.

Route discovery time (RDT) for one route request in TLRDA-D is calculated as:

$$RDT = \begin{cases} 2T_{Class1}h_s(f) & h_s(f) \leq LP_r \\ 2T_{Class1}LP_r + T_{Class2}(h_s(f) - LP_r) + T_{Class1}(h_s(f) - LP_r) & h_s(f) > LP_r \end{cases} \quad (4.20)$$

Where a needed route can be found in two situations: the finder of the route can be within the source node's neighbourhood region where $h_s(f) \leq LP_r$, or the finder of the route can be within beyond-neighbourhood region so $h_s(f) > LP_r$.

End-to-end delay is calculated as follows where *RDT* is calculated in equation (4.20):

$$End - to - end\ delay\ =\ RDT\ +\ T_{Class1} * h_s(d) \tag{4.21}$$

B. Route request latency:

The route request lifetime (*RRL*) is the time the route request resides in the network from the time it is initiated by the source node until it is discarded. Route request is propagated through the network until it is faded or its TTL reaches zero where *D* is the diameter of the network and node *B* is a boundary node.

$$RRL = T_{Class1} LP_r + T_{Class2}(min\ \{D, h_s(B)\} - LP_r) \tag{4.22}$$

The route request latency is the average route request delay per hop so *RRL* is divided by number of hop counts that the route request propagates through the network where *RRL* is calculated in equation (4.22). So Route Request Latency can be calculated as follows:

$$Route\ Request\ Latency\ = RRL\ /min\ \{D, h_s(B)\} \tag{4.23}$$

4.4.1 Comparison between TLRDA-D and AODV

When analysing the delay, we found that the total service time is reduced as a result of the reduction in channel contention for all instances of TLRDA-D. In TLRDA-D, Class 1 and Class 2 packets total service times are reduced by $(C_{max} - C)$ units of time. To illustrate, let us consider the interval of the channel contention to be $0 \leq C \leq 0.8$. In simple flooding, the channel contention is very high which makes $C = 0.8$ while in an ideal network $C = 0$. To count for other factors affecting channel contention, we are assuming that $\acute{C} = 0.2$. Therefore, all AODV packets that use simple flooding belong to Class 1 assuming that $T_{Class1} = 1$ for this algorithm.

In TLRDA-D, when the delay added to route requests increases the channel contention decreases. In TLRDA-d_0, the amount of delay added is small which makes the reduction in channel contention small as well so we will assume $C = 0.7$. Since the delay in TLRDA-d_i is almost double the delay in TLRDA-d_{i-1} when the delay is linear, channel contention in TLRDA-d_i is assumed to be half the channel contention in TLRDA-d_{i-1} so the values for *C* are 0.34, 0.175, and

0.087 for TLRDA-d_1, TLRDA-d_2, and TLRDA-d_3 respectively. Since the delay in TLRDA-d_4 is very large, channel contention is reduced even more where $0 \leq C \leq 0.04$ so $C = 0.02$ is used for TLRDA-d_4. The values of LP calculated as $LP = h_s(f) + 1$. Moreover, Class 2 packet's total service time is calculated according to the values of LP_r because $T_{Class2} = T_{Class1} + \frac{d_i}{(1-\rho)}$ units of time assuming lightly loaded network where $\rho = 0.2$. Furthermore, the hop count is assumed to be the network size divided by 10. Networks are of sizes 20, 30… 100 nodes so hop counts are 2, 3… 10 for different sources and route finders under the same environment.

Figure 4-4 shows that end-to-end delay increases with the increment of network density with all instances of TLRDA-D and AODV. AODV discovers new routes later than all instances of TLRDA-D due to high channel contention. When the delay added to the route request dissemination is increased, the discovery time of the new route is improved because the channel contention is reduced until certain extent.

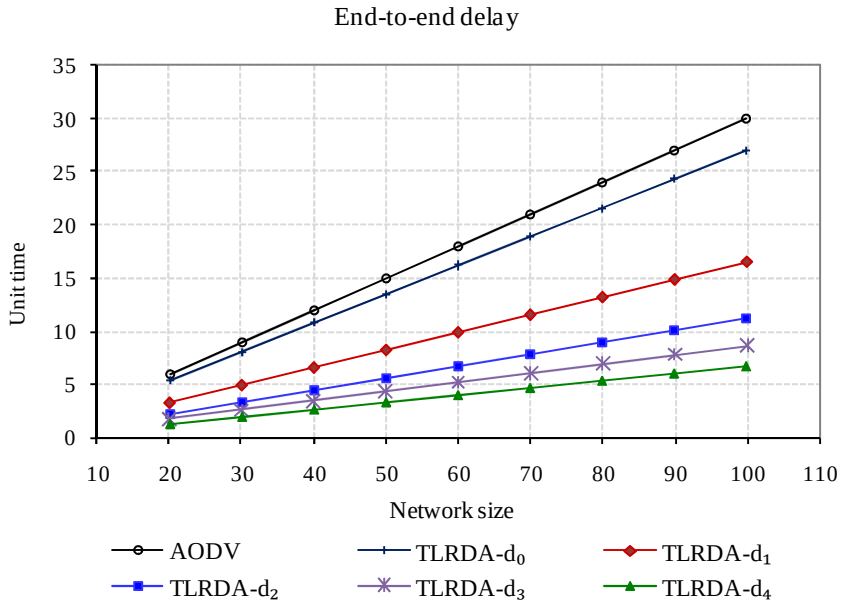

Figure 4-4: End-to-end delay versus network size when $h_s(f) \leq LP_r$.

Due to the delay added to the route request dissemination in the beyond-neighbourhood region, the average of route request latency increases in TLRDA-D more than AODV as shown in Figure 4-5. Moreover, route request latency increases with the increment of the delay added so TLRDA-d_4 gives the highest route request latency among all instances.

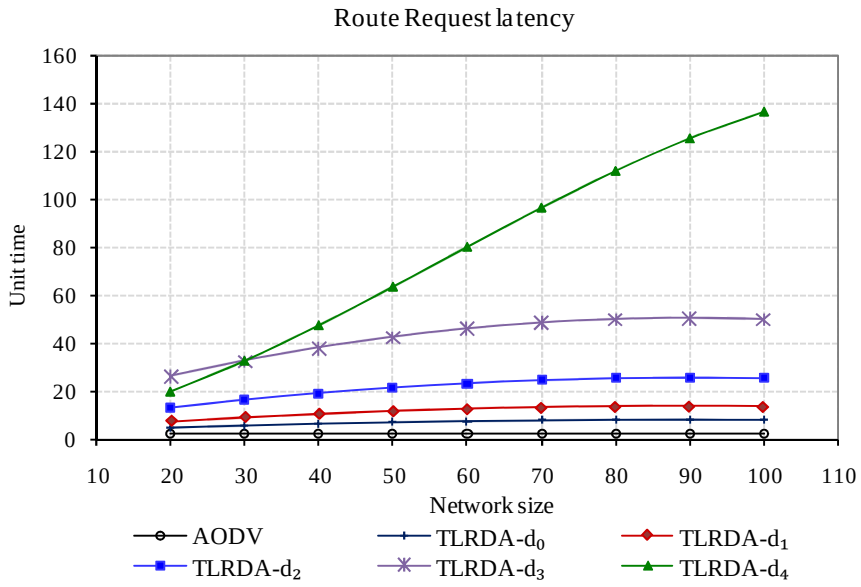

Figure 4-5: Route request latency versus network size when $h_s(f) \leq LP_r$.

4.5 Simulation

A simulation has been conducted to experiment with the new route discovery algorithm, TLRDA-D, and compare it with simple flooding that is used in AODV. TLRDA-D algorithm was implemented as a modification to AODV implementation in ns2, version 2.29 [41]. The comparison focuses on end-to-end delay, route request latency, route request overhead, and packet loss.

The simulation was conducted using five instances of the algorithm, TLRDA-D, corresponding to the different amounts of delay, $d_i, \forall\, i = 0\ to\ 4$ stated earlier in equation (4.1) and in Table 4-1. These runs provide insights and ease the selection of the suitable amount of delay to be used in TLRDA-D. The simulation analysis focuses on the performance of our algorithm, TLRDA-D, as compared to AODV that uses simple flooding from the following prospective: network size, traffic load, and mobility, stated earlier in Table 2-3.

4.5.1 Effect of network size

The network size analysis studies TLRDA-D performance in small to moderate size environment by changing the network size from 20 to 100 as multiple of tens. Nine topologies were run in a squared area of 1000m x 1000m for 900 seconds using RPGM as a mobility model with a

minimum speed 1m/s and a maximum speed of 15m/s where the traffic load is fixed to ten communication sessions. Figure 4-6 to Figure 4-9 display the results of running the five instances of TLRDA-D against AODV.

Latency:

The results reveal that TLRDA-D discovers new routes quicker than AODV. Figure 4-6 shows the superiority of TLRDA-D over AODV especially when d_2, d_3, or d_4 is used as the amount of delay. However, the improvement in the average end-to-end delay is less in small size networks than in moderate size networks. For instance, in TLRDA-d_2, TLRDA-d_3, and TLRDA-d_4, the end-to-end delays were reduced by nearly 52% in small size network and 67% in moderate size network compared to AODV.

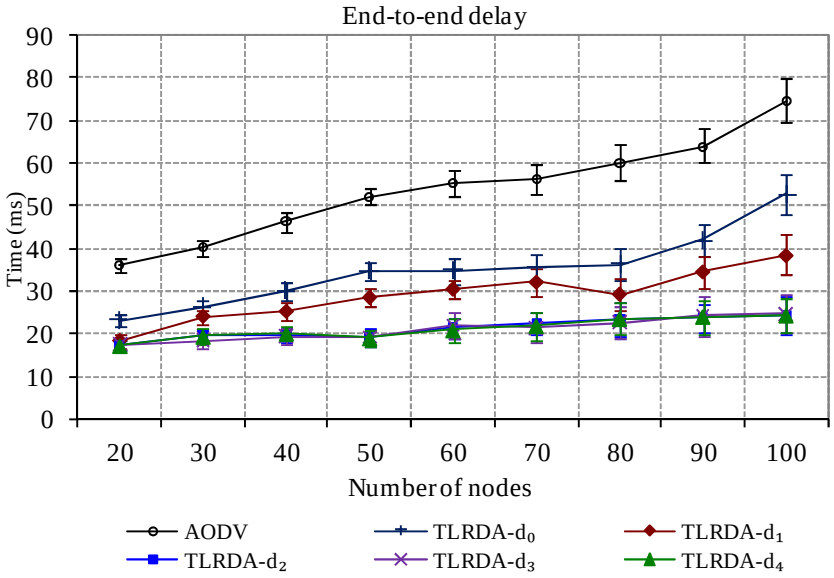

Figure 4-6: End-to-end delay verses network size for networks of 10 communication sessions and 15m/s as maximum speed.

TLRDA-D reduces the average end-to-end delay which enhances the performance by prioritizing the route requests using delays i.e. route requests within the beyond-neighbourhood region have very high chance of being fulfilled already so they are given less priority over other control packets and data including route requests that are being transmitted within their source node's neighbourhood. Such prioritization will minimise channel contention and reduces congestion which improves the average time of route discovery process because it helps other routes to be discovered quickly.

It is worth mentioning that TLRDA-d_4 is the quickest, among all experimental instances of TLRDA-D, in discovering routes as depicted in Figure 4-6 and Table 4-3 which clearly show that d_2, d_3 and d_4 yield in average almost the same end-to-end delay. Figure 4-6 shows that the amount of delay added in TLRDA-d_2 was adequate to achieve the best discovery time in our scenarios as adding more delay will not yield further contention improvement.

In contrast, the latency of route request increases proportionally with the propagation of route requests within the beyond-neighbourhood region. Route requests keep propagating in the network until TTL reaches zero or they fade away. So the route requests in TLRDA-D reside in the network for longer time than in the case of AODV as shown in Figure 4-7 this is due to the added delay which increases overhead yet reduces discovery time. This latency of route request increases with moderate networks in all instances of TLRDA-D, due to the increase of hop count for the same path, but the additional latency is justified when compared to the gain in the route discovery time which reduces end-to-end delay.

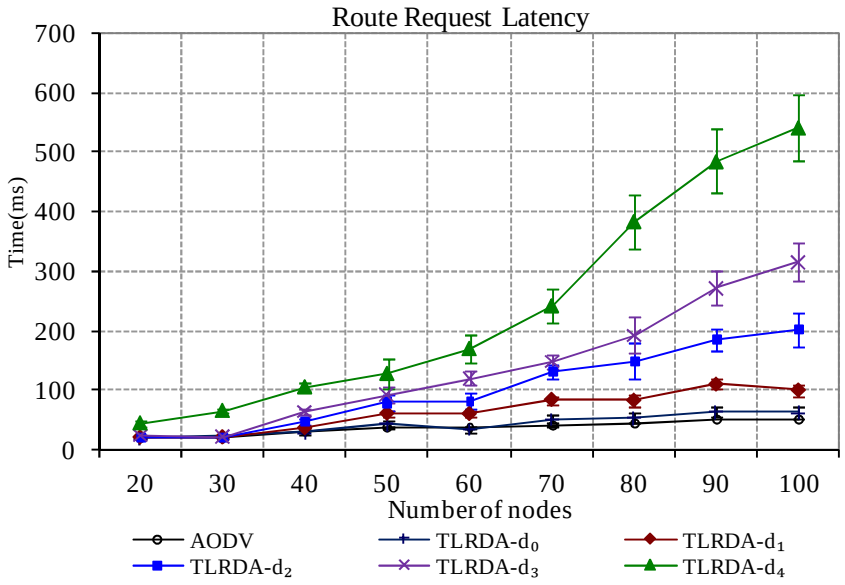

Figure 4-7: Route request latency verses different number of nodes for networks of 10 communication sessions and 15m/s as maximum speed.

Overhead:

The number of transmitted route requests in TLRDA-D and AODV are almost the same regardless of the network size. Due to mobility, the number of transmitted route requests increases or decreases by a very small amount over AODV which slightly increases or decreases the route

request overhead accordingly. Since the two algorithms differ by small amount of transmitted route requests, there are no extra retries of route discovery from source nodes in all experimented instances of TLRAD-D over AODV.

Due to mobility, the longer the route request reside in the network the more the routing breakage happened which may affect the route request overhead as can be depicted in Figure 4-8. Also the reduction of packet loss, Figure 4-9, is one reason behind the increase in the number of received route requests since there are no extra reinitiating attempts for route discovery in TLRDA-D so these route requests have a very high chance of being redundant. The routing overhead increment in TLRDA-D over AODV is small; for instance, in moderate network it ranges up to 21%. However, such small increment is justified compared to the gain in the reduction of end-to-end delay provided by TLRDA-D.

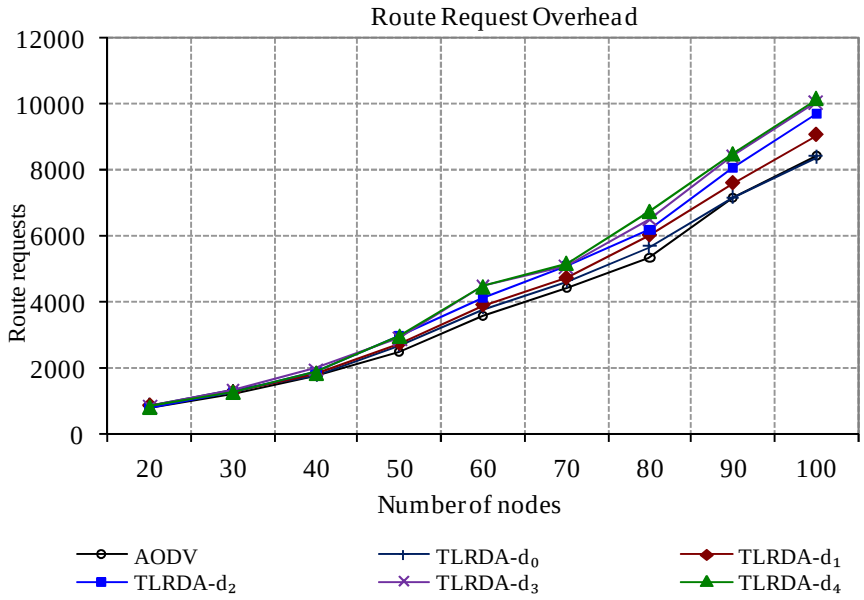

Figure 4-8: Routing overhead verses network size with 10 communication sessions and 15m/s as maximum speed.

Congestion:

TLRDA-D reduces packet loss in the whole network compared to AODV as shown below in Figure 4-9. As the network size increases, TLRDA-D provides better improvement over AODV. The improvement in TLRDA-D over AODV is up to 30% in small size network while it ranges between 22% and 62% in moderate networks as shown in Table 4-3. Some of these packets, gained in TLRDA-D as a result of reducing packet loss, are route requests and this is one reason

behind the increase in route request overhead in TLRDA-D over AODV, as in Figure 4-8, especially knowing the fact that AODV and TLRDA-D have almost the same number of transmitted route requests. In other words, some of the extra route requests received in TLRDA-D are duplicate copies but were dropped because of congestion or/and collision rather than redundancy. The packets that were lost in AODV but gained by TLRDA-D will be referred to as saved packets. To identify the minimum range of saved packets that are not route requests, let us assume that the increase in route request overhead due to the improvement in packet loss only. So, the rest of the saved packets in TLRDA-D (range nearly up to 28% in small size network and 41% in moderate network) can be any kind which might be useful but dropped in AODV due to high channel contention or collision. The saved packets in TLRDA-D improve network performance especially in TLRDA-d_2, TLRDA-d_3, and TLRDA-d_4 because the number of saved packet is larger than TLRDA-d_0 and TLRDA-d_1.

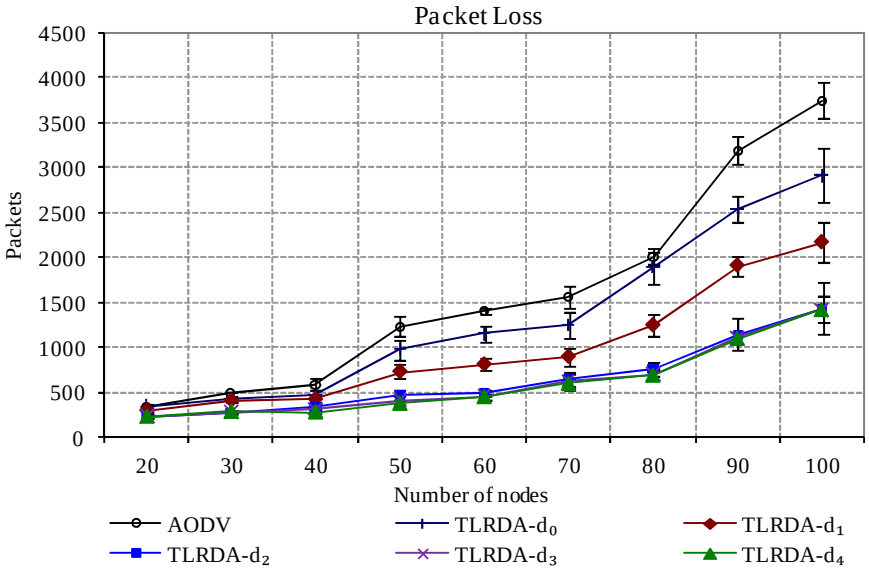

Figure 4-9: Packet loss versus network size with 10 communication sessions and 15m/s as maximum speed.

In summary, TLRDA-D reduces discovery time, packet loss, and end-to-end delay over AODV. However, it increases route request latency and route request overhead in justifiable manner. Furthermore, in TLRDA-d_2, TLRDA-d_3 and TLRDA-d_4 packets propagate with less congestion compared to TLRDA-d_0 and TLRDA-d_1. This improves end-to-end delay and the packet loss leading to better overall network performance, especially in moderate networks, at the cost of higher route request overhead and longer latency which is justifiable.

4.5.2 Effect of traffic load

Figure 4-10 to Figure 4-12 display the results of running the five instances of TLRDA-D against AODV for 900 seconds in an area of 1000m x 1000m. The traffic load is incremented by five starting from 5 to 35 communication sessions to study our algorithm under different amount of traffic loads yet avoid saturation. Network size was fixed at 70 while the random speed ranges between 1m/s and 15m/s.

Latency:

Figure 4-10 demonstrates that the end-to-end delay for TLRDA-D is less than that of AODV in spite of traffic load. However, the end-to-end delays for TLRDA-d_2, TLRDA-d_3, and TLRDA-d_4 are less than the time for both TLRDA-d_0 and TLRDA-d_1. Furthermore, the end-to-end delay for TLRDA-d_i where $i > 0$ is further improved with heavy traffic load because when the communication sessions are increased the number of route requests needed is also increased when the destinations is unknown to the sender.

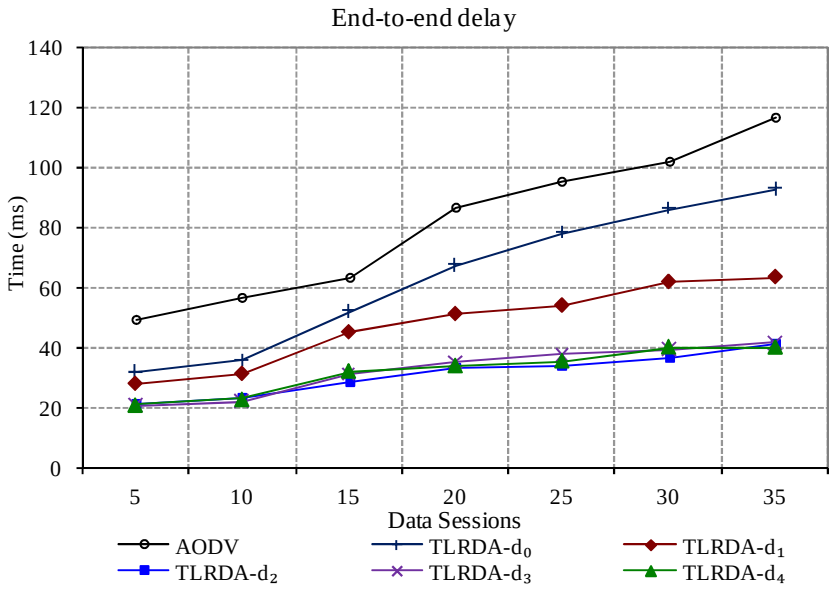

Figure 4-10: End-to-end delay versus traffic load with a network70 nodes and 15m/s as maximum speed.

Also in this analysis, when TLRDA-D uses d_2, d_3 or d_4 as amount of delay, the algorithm yield in average almost the same the end-to-end delay as depicted from Figure 4-10 for these three instances among all experimented instances of TLRDA-D where the end-to-end delay was reduced by nearly 57% in light traffic and 65% in heavy traffic for TLRDA-d_2, TLRDA-d_3, TLRDA-d_4

compared to AODV. So, TLRDA-D has end-to-end delay lower than AODV from traffic load prospective. Furthermore, TLRDA-d_2, TLRDA-d_3, and TLRDA-d_4 have almost the same end-to-end delay that is lower compare to both TLRDA-d_0 and TLRDA-d_1. This improvement in the end-to-end delay is due to the reduction in channel contention thus the data can travel earlier and quicker which improves the network performance.

Route request latency increases with the increment of traffic load in TLRDA-D as shown in Figure 4-11. Furthermore, when the delay is less than polynomial the route request latency increases slightly with the increment of d_i where $i = 0, 1, 2, or\ 3$. Otherwise, the route request Latency increases in a larger amount, as in TLRDA-d_4, especially under heavy traffic. As we mentioned before, the delay added to the fulfilled route requests do not affect the discovery process because it is added within the beyond-neighbourhood region.

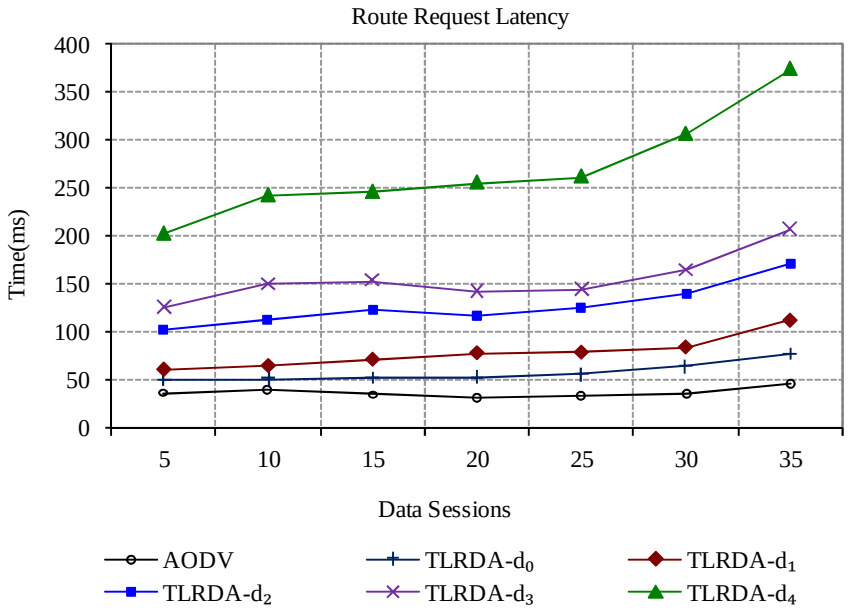

Figure 4-11: Route request latency versus traffic load with a network size of 70 nodes and 15m/s as maximum speed.

Overhead:

Also in this analysis, some of these saved packets in TLRDA-D might be route requests which justify the increment in route request overhead in TLRDA-D over AODV as in Figure 4-12. AODV and TLRDA-D have almost the same number of transmitted route request so any extra route request is most likely to be a duplicate and thus will be dropped any way. Furthermore, the number of saved packets is greater than the increment in route request overhead in TLRDA-D. The

difference represent the minimum number of saved packets that is not duplicate route requests where those saved packets might be useful and range from 27% to 45% in light traffic and 26% to 36% in heavy traffic. Those saved packets can be any kind of packets which might be useful but dropped in AODV due to contention, congestion, or collision. TLRDA-D improves network performance by utilising the useful saved packets.

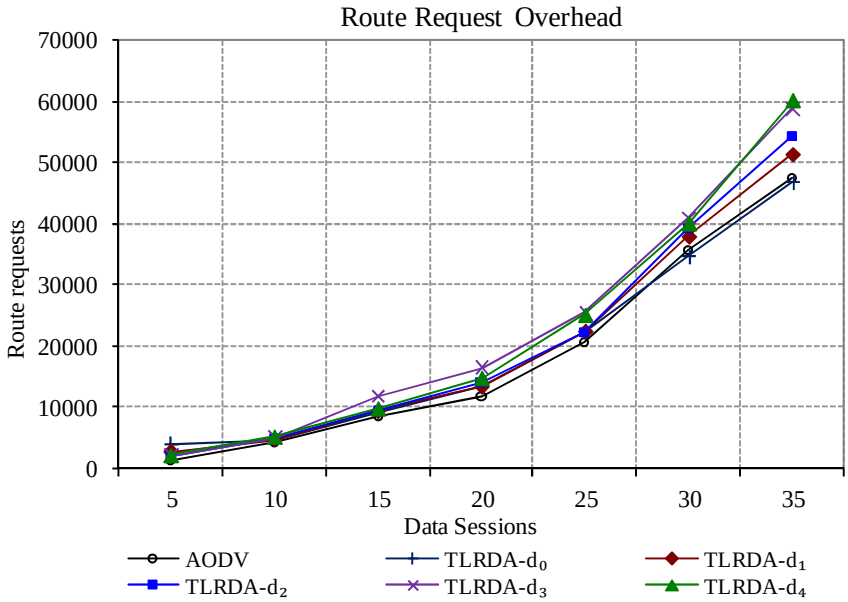

Figure 4-12: Routing overhead versus traffic load in a network of 70 nodes and 15m/s as maximum speed.

Congestion:

Moreover, TLRDA-D reduces packet loss in the whole network compared to AODV as shown below in Figure 4-13. The improvement in TLRDA-D over AODV ranges from 49% to 65% in light traffic while it ranges between 34% and 53% in heavy traffic. The packet loss is nearly the same for TLRDA-d_2, TLRDA-d_3, and TLRDA-d_4. Furthermore, the reduction in packet loss in these three instances is better than in TLRDA-d_0 and TLRDA-d_1.

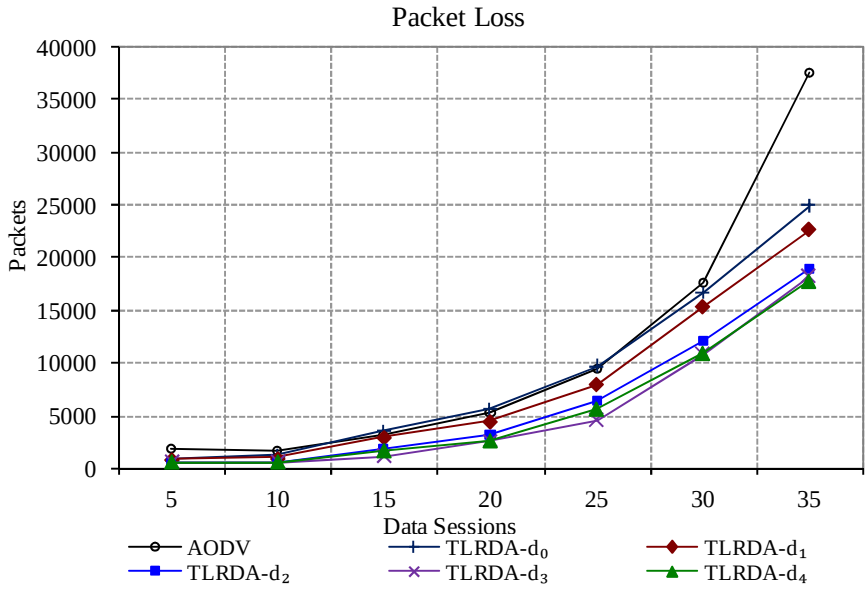

Figure 4-13: Packet loss versus traffic load in networks of 70 nodes and maximum speed of 15m/s.

Traffic load analysis conducted in the five instances of TLRDA-D shows that TLRDA-d_2 achieves the best end-to-end delay, and packet loss compared to TLRDA-d_0 and TLRDA-d_1 with little overhead and lower route request latency compared to TLRDA-d_3 and TLRDA-d_4.

4.5.3 Effect of mobility

Figure 4-14 to Figure 4-16 were derived from simulating the five instances of TLRDA-D and AODV while the maximum speed increases by taking one the following values: 2, 5, 7, 10, 13, and 15m/s in a network of size 70 nodes with 10 communication sessions.

Latency:

The end-to-end delay in TLRDA-D is reduced compared to AODV for different maximum speed as in Figure 4-14 where discovery time increases in both TLRDA-D and AODV with fast speed because speed affects routes and may result in broken links. This figure reveals the difference in the end-to-end delay among all five instances of TLRDA-D where TLRDA-d_2, TLRDA-d_3, and TLRDA-d_4 reduce the end-to-end delay more than TLRDA-d_0 and TLRDA-d_1. So, this figure shows that when TLRDA-d_2, TLRDA-d_3, and TLRDA-d_4 are used, the algorithm yield almost the same end-to-end delay as depicted from Figure 4-14.

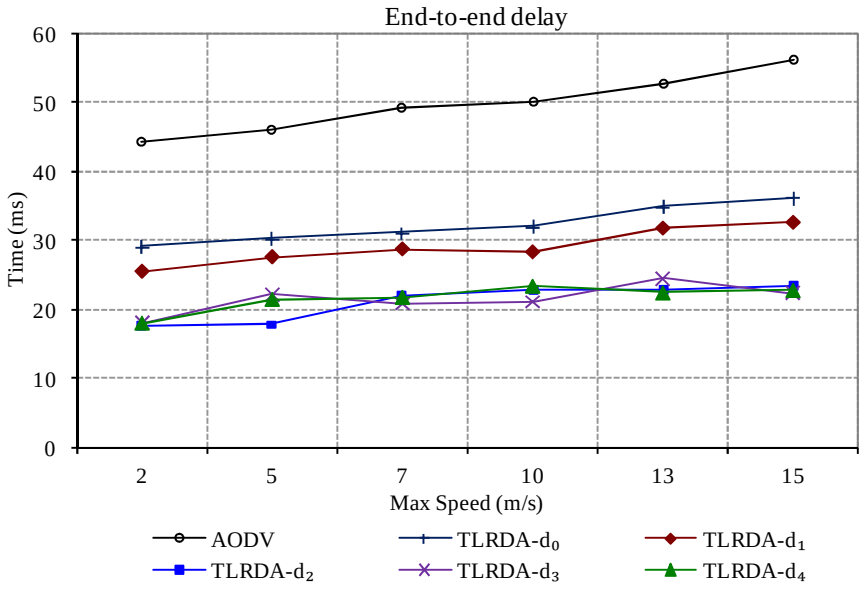

Figure 4-14: End-to-end delay versus maximum speed in networks of 70 nodes and 10 communication sessions.

Route requests in TLRDA-D tend to stay active in the network longer than AODV as in Figure 4-15. Furthermore, the route request latency increases slightly when the delay is less than polynomial. Otherwise, the route request latency increases in a larger amount, as in TLRDA-d_4, where the delay is polynomial.

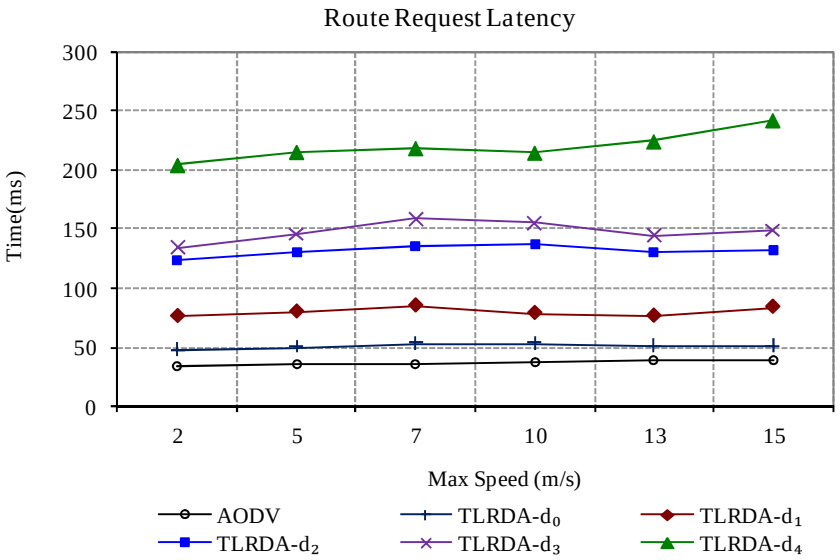

Figure 4-15: Route request latency versus maximum speed, 70 nodes, and 10 communication sessions.

Overhead:

Both AODV and TLRDA-D have almost the same number of transmitted route request; so the extra route requests received in TLRDA-D as shown in Figure 4-16 might be duplicate copies but

were dropped because of congestion or collision. Furthermore, the number of saved packets is greater than the increment in route requests overhead where the minimum difference ranges up to 70% in slow speed and up to 45% in fast speed. The extra saved packets can be any kind of packets which might be useful but dropped in AODV due to any reason i.e. contention, congestion, or collision. These saved packets in TLRDA-D have a good impact on network performance, as mentioned before in network size and traffic load analyses.

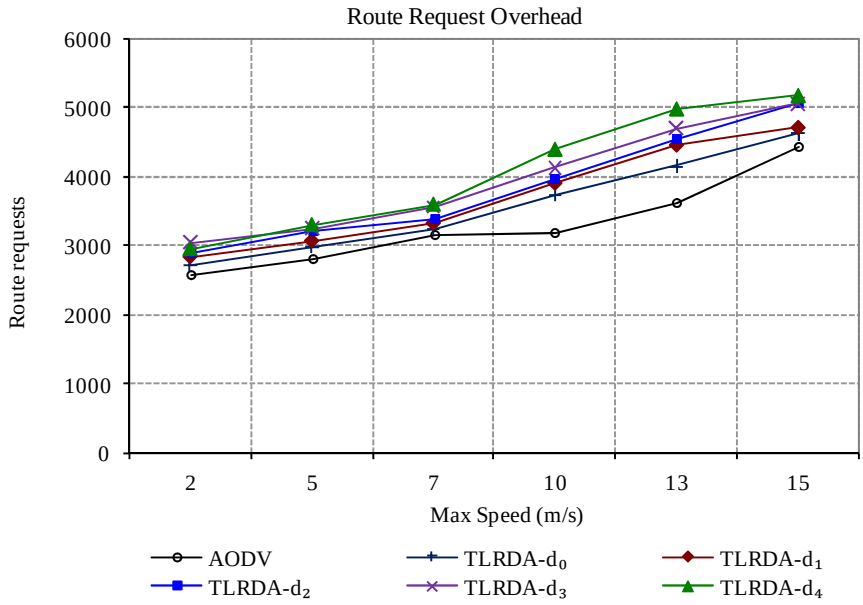

Figure 4-16: Routing overhead versus maximum speed with networks of 70 nodes and 10 communication sessions.

Congestion:

Also in this analysis, TLRDA-D reduces packet loss in the whole network compared to AODV as shown below in Figure 4-17. Packet loss increases with faster movements in both algorithms. TLRDA-D improves packet loss over AODV by up to 87% in slow speed and from 21% to 62% in fast speed. Moreover, these packets include route requests which increases route request overhead in TLRDA-D over AODV as in Figure 4-16.

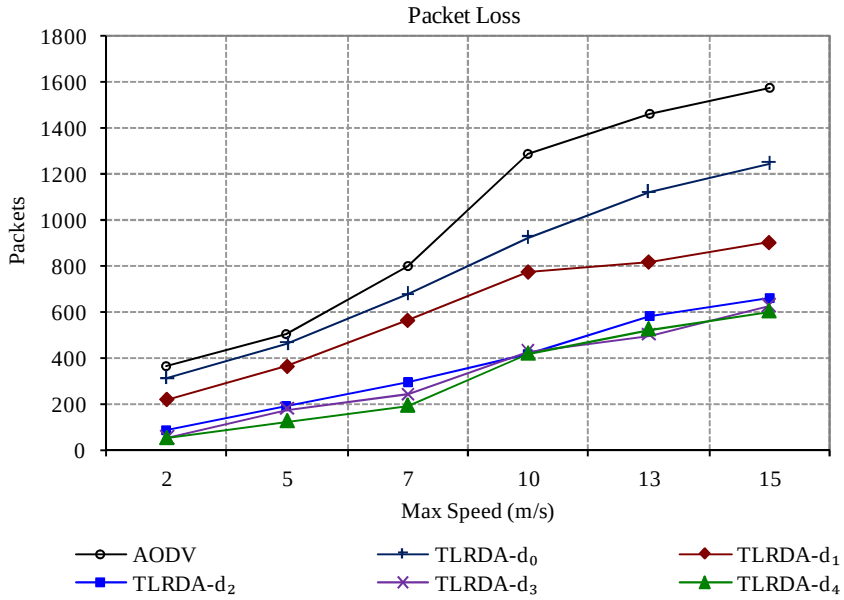

Figure 4-17: Packet loss versus maximum speed with 70 nodes and 10 communication sessions.

4.6 Summary of the simulation results

Simulation experiments have been conducted to study the performance of TLRDA-D when the network size, traffic load, and mobility are varied. The results of these three analyses have revealed almost the same relative performance behaviour between the new TLRDA-D and AODV is observed with respect to for the following metrics: end-to-end delay, packet loss, route request latency, and route request overhead as depicted in Figure 4-6 to Figure 4-16. Therefore, delaying the propagation of route requests after discovering the needed route reduces network congestion which improves the discovery time of needed routes and reduces packet loss.

Our simulation has considered five values for the delay parameter in TLRDA-D. This delay is defined to be a monotonically non-decreasing function of the locality parameter (*LP*) with logarithmic, linear, or polynomial increase. Our objective is to narrow down the best value for the delay function for the considered scenarios. Moreover, Table 4-3 presents the comparison between the experimented instances of TLRDA-D and AODV.

TLRDA-D discovers routes quicker when the amount of delay is larger than logarithmic thus the logarithmic increase was ruled out. TLRDA-d_2, TLRDA-d_3, and TLRDA-d_4 reduce end-to-end delay almost by the same amount but TLRDA-d_4 yields high route request latency thus the polynomial increase was ruled out too. Hence, the best delay function would be a linear one. In

particular, for the considered scenarios in our experimental study the doubling function where $d_2 = 2LP_r$ provides the best among all scenarios therefore it is the turning point. For our scenarios, TLRDA-d_2 is chosen to be the best among the five instances of TLRDA-D because it achieves low end-to-end delay and packet loss with less increment in route request latency and routing overhead.

Table 4-3: Percentage of changes in all five instances of TLRDA-D over AODV.

cases	Algorithm	End-to-end delay (reduction)		Packet Loss (reduction)		Route request latency (increase)		Routing Overhead (increase)	
Network size		Small	Moderate	Small	Moderate	Small	Moderate	Small	Moderate
	TLRDA-d_0	36%	29%	1%	22%	0%	28%	0%	3%
	TLRDA-d_1	49%	49%	11%	42%	7%	102%	3%	8%
	TLRDA-d_2	52%	67%	30%	62%	13%	305%	0%	15%
	TLRDA-d_3	52%	67%	30%	62%	25%	537%	1%	19%
	TLRDA-d_4	52%	67%	30%	62%	137%	990%	2%	21%
Traffic load		Light	Heavy	Light	Heavy	Light	Heavy	Light	Heavy
	TLRDA-d_0	36%	20%	49%	34%	39%	67%	16%	2%
	TLRDA-d_1	45%	46%	52%	40%	68%	145%	23%	8%
	TLRDA-d_2	57%	65%	65%	50%	184%	271%	27%	14%
	TLRDA-d_3	58%	64%	65%	51%	248%	350%	48%	24%
	TLRDA-d_4	58%	66%	65%	53%	464%	713%	56%	26%
Mobility		Slow	Fast	Slow	Fast	Slow	Fast	Slow	Fast
	TLRDA-d_0	34%	36%	14%	21%	14%	30%	6%	5%
	TLRDA-d_1	43%	42%	40%	43%	123%	112%	11%	7%
	TLRDA-d_2	60%	58%	77%	58%	263%	233%	13%	15%
	TLRDA-d_3	59%	60%	87%	60%	293%	276%	15%	14%
	TLRDA-d_4	59%	59%	86%	62%	498%	511%	15%	17%

4.7 Conclusions

The Traffic Locality oriented Route Discovery Approach (TLRDA), introduced in Chapter 3, is utilised to establish then maintain each node neighbourhood in an effort to improve the route discovery process through the development of and Traffic Locality oriented Route Discovery Algorithm with Delay (TLRDA-D). It works by adding a delay to route request dissemination in the beyond-neighbourhood region with respect to the source node which initiates the route request to reduce channel contention which reduces the discovery time of other route requests. The

simulation analysis has shown that when TLRDA-D uses twice the locality parameter (*LP*) as a delay it gave the best improvement among the examined scenarios thus will be used as the amount of delay for TLRDA-D in the next chapter. TLRDA-D improves the end-to-end delay and reduces packet loss regardless of network size, traffic load, or maximum speed.

Chapter 5: Traffic Locality oriented Route Discovery Algorithm with Chase

5.1 Introduction

The new route discovery algorithms described in this research are aimed for MANET applications that exhibit traffic locality. The approach TLRDA that was introduced in Chapter 3 earlier, establishes neighbourhood and beyond-neighbourhood regions for each active source node. The neighbourhood region includes most of the likely destinations for a given source node. Route request is broadcast without adding any delay within its source node's neighbourhood boundary to discover routes quickly. TLRDA-D, proposed and studied in Chapter 4, reduces the network end-to-end delay but with a justifiable price of increasing the routing overhead which in turn requires more resources. This is because route request resides in the network even if it is unneeded until it fades away when reaching a boundary node or discarded because its TTL field reaches zero.

In this chapter, a new traffic locality oriented route discovery algorithm with chase, TLRDA-C, is introduced that uses chase packets to limit the propagation of route request and overcome the problem of TLRDA-D. The chasing idea has been used in Limited Broadcasting [44] and has been applied later to enhance ERS in Blocking-ERS [89, 90]. It works by trying to free the network from the unneeded route requests. Chase packet is a broadcast control packet which is disseminated through the network after discovering the route to discard the fulfilled route request.

5.1.1 Limited Broadcasting

Limited Broadcasting [44] (L-B for short) improves the route discovery process by using chase packets to stop the fulfilled route request packets from further propagation after finding the required route. The algorithm works by creating two virtual channels as an abstract division of the time slots available; in a bidirectional network with unknown distance between source and destination. It uses these virtual channels to divide time among route requests, route replies, and chase packets. Moreover, the first channel uses one time slot, ¼ of the time, while the second

channel uses the rest of the time slots i.e. ¾ of the time. A node in L-B broadcasts route requests using the first channel while the second channel is used to transmit the route replies or broadcast chase packets.

The main deficiency of the L-B algorithm is that it favours the chase packets and route replies over the route requests from the start. Route requests are delayed from the start before discovering the needed route which would delay all route discoveries. Delaying the discovery process might reduce the chance of finding new routes while delaying route replies might increase the possibility of losing these routes after their discovery and may hinder the discovery and the chasing processes.

5.1.2 Blocking-ERS

Blocking-ERS (B-ERS) [89, 90] improves energy consumption by stopping the fulfilled route requests. B-ERS uses chase packets to stop the propagation of route requests after discovering the required route. It is an improvement of the Expanding Ring Search (ERS) [115] where each new ring starts from the previous ring instead of starting from the source node as in ERS. B-ERS works by introducing a delay equal to *2hop-count*NTT* at each ring where rings are increased sequentially. After this delay, the intermediate nodes in the current ring may receive a chase packet called "stop_instruction" from the source node. Stop_instruction is broadcast to cover the current ring only where the finder of the route is located. Upon receiving the chase packet, the intermediate node will discard both the route request and the chase packet. If no chase packets are received within $2hop\text{-}count * NTT$ units of time, the node will rebroadcast the route request to cover a larger ring. Chase packet is broadcast up to $h_s\,(f)$ distance at maximum to cover only the ring in which the finder of the route resides. The source node needs to know how many hops away does the finder of the route reside, thus the format of route reply packet should be extended by one byte to carry the value of $h_s\,(f)$.

The two main deficiencies of the B-ERS algorithm are: first, it delays the route request from the start where the route is not discovered yet which increases the end-to-end delay and might reduce the chance of finding new routes. Second, nodes in B-ERS broadcast chase packets to cover only the ring where the finder of the route resides at the time of discovery. In the presence of mobility,

this restriction may hinder the chasing process and reduces the success rate of the catching mechanism.

5.2 The Traffic Locality oriented Route Discovery Algorithm with Chase (TLRDA-C)

We propose a new algorithm called TLRDA-C that utilises TLRDA and TLRDA-D, introduced in Chapter 3 and Chapter 4, in addition to the chase packet concept. Since TLRDA-C is designed for applications that exhibit traffic locality in MANETs, TLRDA is used to establish neighbourhood and beyond-neighbourhood regions for each active source node. A node in TLRDA-C algorithm, as in TLRDA-D, broadcasts route request within the neighbourhood region according to the routing algorithm used. Afterwards, it broadcasts the route request with a delay equal to $2LP_r * NTT$ outside such neighbourhood.

TLRDA-C is an improvement of TLRDA-D as an effort to reduce the route request latency and to improve routing overhead whilst keeping the route discovery time low. The main idea of TLRDA-C is to process the route discovery fast within the neighbourhood boundary, as it would cover most of the destinations. However, the route request would slowdown and continues at the same speed as it propagates in the beyond-neighbourhood boundary to reduce contention and to give the chasing mechanism a better chance to succeed. The source node is informed about the discovery of the required route by the route reply which implies that the discovery process should be stopped. The source node transmits a chase packet to inform other intermediate nodes about this discovery in order to stop broadcasting the fulfilled route request. The chase packet is broadcast without adding any delay in an effort to terminate the propagation of the fulfilled route request as soon as possible. The catching occurs in the beyond-neighbourhood region as the chase packet travels faster than its associated route request within this region.

Figure 5-1 shows the steps that are performed by each node upon receiving a route request where the first step is to discard any duplicate route requests (line 2). If the route request received for the first time (line 3), the node searches the stored information for the matching chase packet, if found (line 4) the route request will be discarded after storing the needed information (lines 5-6). If no matching chase packet was received, the node stores the route request for double the LP_r units of

time (line 9) if the performing node resides in the route request source node's beyond-neighbourhood region. Otherwise, if the performing node resides in the source node's neighbourhood region, the route request is processed according the routing protocol used (line 11).

Steps preformed by each node upon receiving a route request in TLRDA-C

```
1:  If route request is a duplicate
2:      Discard the route request
3:  Else
4:      If chase packet has been received then
5:          Store route request information
6:          Discard the route request
7:      Else
8:          If hop_count > LP_r then
9:              Wait (2LP_r) unit time
10:         End if
11:         Process the route request
12:     End if
13: End if
```

Figure 5-1: Processing of route requests at a node in TLRDA-C.

When a route reply is received, the receiving node performs the steps in Figure 5-2. If the receiving node is the source node (line 1), it creates the associated chase packet then broadcasts it (lines 2-3). After that, the source is ready to start transmitting the actual data (line 4). The last step (line 6) is performed by all nodes to process the route reply according to the routing protocol used.

Steps performed by each node upon receiving a reply packet in TLRDA-C

```
1:  If current node is the sender then
2:      Create a chase packet
3:      Broadcast the chase packet
4:      Start transmitting the data
5:  End if
6:  Process the route reply
```

Figure 5-2: Processing of route replies at a node in TLRDA-C.

Upon receiving the chase packet, the steps in Figure 5-3 are performed at each node. If the chase packet is a duplicate, it is discarded by the node (line 2). Otherwise, the needed information is stored (line 4) where each node keeps track of all received route requests and chase packets for BCT units of time by storing the needed information i.e. their broadcast ID and originator IP

address. If route request and chase packet information stored in the same table, a bit flag is needed to distinguish between route requests and chase packet records. If the matching route request is broadcast already then the chase packet is broadcast as well (line 7) but if the route request is waiting to be broadcast then both the route request and its matching chase packet are discarded (line 9). If the route request is not received yet, the chase packet is discarded (line 12) after storing the needed information (line 4).

Steps performed by each node upon receiving the chase packet in TLRDA-C

```
1:  If the chase packet is duplicate then
2:      Discard it.
3:  Else
4:      Store chase information
5:      If the route request is received then
6:          If the route request is broadcast then
7:              Broadcast the chase packet.
8:          Else
9:              Discard both packets.
10:         End if
11:     Else
12:         Discard the chase packet.
13:     End if
14: End if
```

Figure 5-3: Processing of chase packets at a node in TLRDA-C.

TLRDA-C implements the mechanism as in TLRDA for updating its neighbourhood boundary using the most recent routes discovered for that source node so the boundary will be dynamically changing as the network status changes. If the destination is beyond the neighbour boundary it will be eventually discovered without the need for any boundary immediate expansion strategy because the route request will be travelling outside its boundary but with a slower speed.

In TLRDA-C, the source node is always the initiator of the chase packets regardless of the routing method in place whether it is uni-path as in DSR and AODV or multi-path as in Ad-hoc On-demand Multipath Distance Vector protocol (AOMDV) [75] and Multi-Path Dynamic Source Routing protocol (MP-DSR) [69]. This enables TLRDA-C to avoid initiating many chase packets for the same route request; at the cost of tiny amount of delay equals to $h_s(f)$. In the case of

multi-path routing protocols, the sender needs to discover additional routes for the same destination as backups. So the sender is the only node that observes the discovery of all the required routes. As a result, the sender initiates the chase packet as soon as it knows that such routes are discovered; this happens immediately upon receiving the route reply(s).

TLRDA-C assumes that the route finder, f, is not located near the boundaries of the network which is mostly the case; otherwise the chase packet may be unable to catch the route request leading to a situation where the overhead will overcome the benefits.

5.2.1 Chase Packet Format

Packet size should be chosen carefully because transmitting and receiving consumes bandwidth and power in wireless networks. In MANETs the packets cross multiple nodes, thus using a small packet size is more efficient in a resource-wise manner across the network. So chase packets in TLRDA-C are kept small in size, 16 bytes, compared to a route request packet in order to minimise resources consumption. The route request sizes in TLRDA-C and AODV are 25 and 24 bytes respectively. The format of chase packet is shown in Figure 5-4 and the fields of the chase packet are described in Table 5-1.

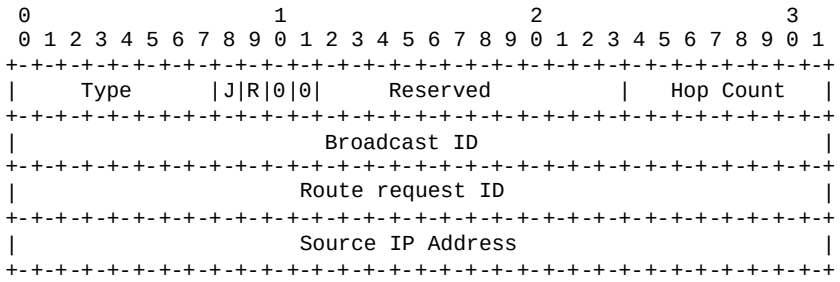

Figure 5-4: The format of chase packet.

The route request ID and the source IP address uniquely identify the particular route request that is associated with the chase packet while the broadcast ID and the source IP identify a unique chase packet.

Table 5-1: Description of the fields for the chase packet format.

Field name	Field description
Type	5 (CHASE)
J, R and Reserved	Reserved for future use i.e. multicast.
Hop Count	The number of hops from the Source to the current node.
Broadcast ID	Chase packet broadcast ID.
Route request ID	Route request broadcast ID.
Source IP Address	The IP address of the source node.

5.3 Mathematical formulation

To simplify the mathematical formulation we will not consider the role of mobility in this section. When a route finder f, at distance $h_s(f)$ from the source s, sends a route reply to the source in the reverse direction it discards that route request. However, other nodes will continue to broadcast the route request throughout the network since they may not be aware of the successful route discovery by node *f*.

When the source node receives the route reply, it initiates and broadcast the chase packet while the route request still propagating throughout the network. Let us assume that the route request is twice the distance from the source when the reply reaches the source node i.e. $2h_s(f)$. Moreover, by the time the chase packet is $2h_s(f)$ distance from the source; the route request would have propagated further and the chase packet would still be chasing it.

In this section, we are modelling TLRDA-C with respect to delay. Let us consider a route request that is chased by a chase packet travelling in the same direction. Let the speed of both the route request and the chase packet be v_1 and the total service time per a node of a chase packet and a route request travelling within its neighbourhood region be T_{Class1}; while the total service time for a route request travelling within its beyond-neighbourhood region be T_{Class2}. Therefore, within the neighbourhood region the route request and the chase packet are experiencing the same total service time T_{Class1} where T_{Class1} and T_{Class2} were derived earlier when modelling TLRDA-D in Section 4.3.

When the chase packet is initiated, there will be a distance of $2h_s(f)$ between the route request

and the chase packet. Furthermore, the chase packet will always catch the route request in the beyond-neighbourhood region where $T_{Class1} < T_{Class2}$; otherwise, when $T_{Class1} \geq T_{Class2}$ chase packets experience the same or more delay than route requests, the catching process is imposable.

Below we will calculate the route discovery time (RDT) and the route request lifetime (RRL). There are only two possibilities to be considered which are stated as Case 1 and Case 2:

- Case 1: The route request is in the neighbourhood region at time (t) when the source initiates and broadcasts the chase packet i.e. $h_s(f) \leq \frac{LP_r}{2}$.
- Case 2: The route request will be in the beyond-neighbourhood region at time (t) when the source initiates and broadcast the chase packet. i.e. $h_s(f) > \frac{LP_r}{2}$.

A. Calculating the route request lifetime (RRL)

The route request lifetime *(RRL)*, the total broadcast time, is the time from sending a particular route request until the chase packet catches such route request and causes it to be discarded. So we need to calculate the chasing time (t_c) first. The chase packet will cause the route request that is associated with it to be discarded at $t_c v_1 T_{Class1}$ distance away. To calculate the chase time (t_c) in both cases, let us define the distance travelled by one route request and its chase packet within beyond-neighbourhood to be $t_{rc} v_1 T_{Class2}$ and $t_{rc} v_1 T_{Class1}$ respectively at speed of v_1. When the chase packet is initiated by the source node its route request will be $2h_s(f)$ away in all directions simultaneously. In TLRDA-C, all chase packets and route requests in their neighbourhood region belong to Class 1 while route requests in their beyond-neighbourhood region belong to Class 2.

The time t_c that is needed for a particular chase packet to catch the route request associated with it can be calculated using the following formula:

$$t_c = LP_r T_{Class1} + \mathrm{t_{rc}} \tag{5.1}$$

Let us consider Case 1, when route request is within neighbourhood region at time t that is after travelling $2h_s(f)$ distance by the route request as shown in Figure 5-5.

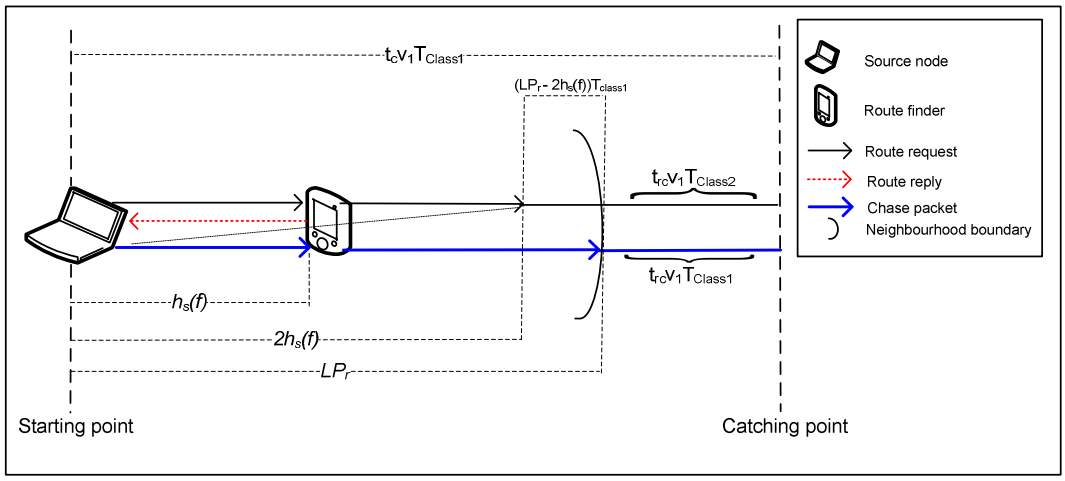

Figure 5-5: Illustration of Case 1.

The value of t_{rc} can be calculated from the following formula:

$$t_{rc} v_1 T_{Class2} + \left(LP_r - 2h_s(f)\right) T_{Class1} = LP_r T_{Class1} + t_{rc} v_1 T_{Class1} \quad (5.2)$$

By simplifying equation (5.2):

$$t_{rc} v_1 \left(T_{Class2} - T_{Class1}\right) = 2h_s(f)\, T_{Class1} \quad (5.3)$$

Giving the value of t_{rc} as follows:

$$t_{rc} = \frac{2h_s(f) T_{Class1}}{v_1 (T_{Class2} - T_{Class1})} \quad (5.4)$$

To get the value of the chase time t_c for Case 1, equation (5.4) is substituted in equation (5.1) as follows:

$$t_c = LP_r T_{Class1} + \frac{2h_s(f) T_{Class1}}{v_1 (T_{Class2} - T_{Class1})} \quad (5.5)$$

The route request lifetime (RRL) is calculated as:

$$RRL = 2h_s(f) T_{Class1} + t_c \quad (5.6)$$

Using (5.5) and (5.6), *RRL* becomes:

$$RRL = 2h_s(f)T_{Class1} + LP_rT_{Class1} + \frac{2h_s(f)T_{Class1}}{v_1(T_{Class2} - T_{Class1})} \quad (5.7)$$

Now let us consider Case 2, when route request within beyond-neighbourhood region at time t. As illustrated in Figure 5-6.

We can calculate t_{rc} from the following formula:

$$t_{rc}v_1 T_{Class2} - (2h_s(f) - LP_r) T_{Class2} = LP_rT_{Class1} + t_{rc}v_1T_{Class1} \quad (5.8)$$

And by simplifying this equation, we get:

$$t_{rc}v_1 (T_{Class2} - T_{Class1}) = 2h_s(f) T_{Class2} + LP_rT_{Class1} - LP_r T_{Class2} \quad (5.9)$$

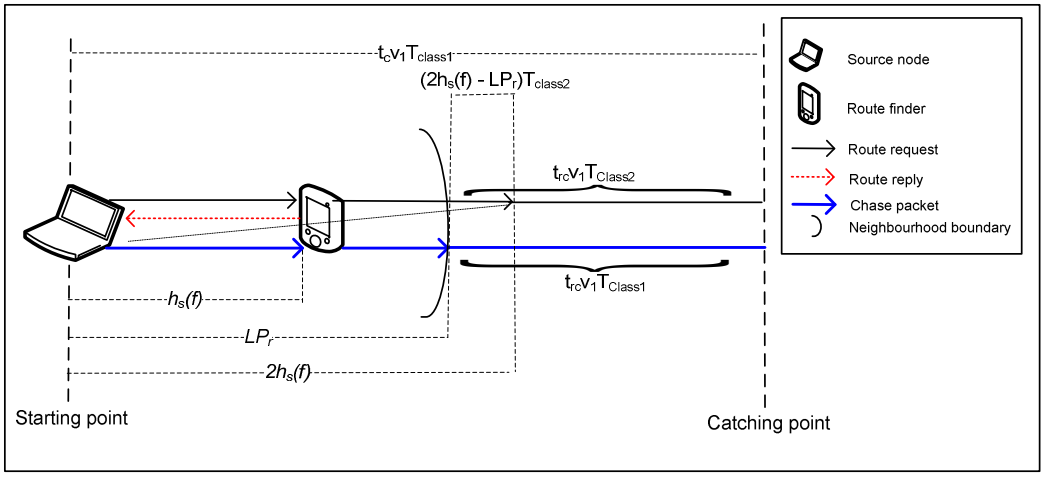

Figure 5-6: Illustration of Case 2.

This gives us the value of t_{rc} as follows:

$$t_{rc} = \frac{2h_s(f) T_{Class2} + LP(T_{Class1} - T_{Class2})}{v_1(T_{Class2} - T_{Class1})} \quad (5.10)$$

The chase time t_c for Case 2 can be calculated by substituting t_{rc} in equation (5.1) by its value from equation (5.10) as follows:

$$t_c = LP_r T_{Class1} + \frac{2h_s(f)\,T_{Class2} + LP(T_{Class1} - T_{Class2})}{v_1(T_{Class2} - T_{Class1})} \quad (5.11)$$

The route request lifetime (RRL) can be calculated as follows:

$$RRL = (2h_s(f) - LP_r)T_{Class2} + LP_r T_{Class1} + t_c \quad (5.12)$$

By substituting t_c from (5.11) in (5.12):

$$RRL = 2h_s(f)\,T_{Class2} + LP_r(T_{Class1} - T_{Class2}) + LP_r T_{Class1} + \frac{2h_s(f)\,T_{Class2} + LP_r(T_{Class1} - T_{Class2})}{v_1(T_{Class2} - T_{Class1})} \quad (5.13)$$

B. Calculating the route request latency

The average of route request latency per hop can be calculated by dividing the route request lifetime RRL by the number of hop counts that the route request traverse. RRL depends on the case used.

$$Route\ request\ latency = \frac{\mathrm{RRL}}{t_c v_1} \quad (5.14)$$

C. Calculating the route discovery time (RDT)

The route discovery time (RDT) is the round trip time from sending a particular route request by the source node until it receives the first route reply.

For Case 1, when the finder of the required route is within the neighbourhood region at time t, the route discovery time (RDT) is calculated as:

$$RDT = 2h_s(f)T_{Class1} \quad (5.15)$$

For Case 2, when the finder of the required route is within beyond-neighbourhood region, RDT is calculated as:

$$RDT = h_s(f)T_{Class1} + LPT_{Class1} + (h_s(f) - LP)T_{Class2} \quad (5.16)$$

D. Calculating end-to-end delay

The end-to-end delay can be calculated by adding the route discovery time to the time the data packet needs to reach the destination, assuming that data packets have total service time equal to T_{Class1}. RDT depends on the finder of the route being within the neighbourhood or not i.e. Case 1 or Case 2.

$$End - to - end\ delay\ = RDT + h_s(d)T_{Class1} \tag{5.17}$$

5.3.1 Comparison with existing algorithms

In this subsection we conduct a comparison between TLRDA-C, Limited Broadcasting [44], and Blocking-ERS [89, 90] using various values of hop counts for independent route discoveries with different sources and route finders under the same environment.

TLRDA-C is compared with both L-B and B-ERS algorithms to evaluate the trends of the route discovery time (RDT) and the route request lifetime (RRL) against those of L-B and B-ERS. In TLRDA-C, RRL and t_c metrics are related to each other because the chase packet needs to travel the same distance as the associated route request for all chase packets to succeed in the catching process.

In this comparison, the speed of route request or chase packet v_1 is 1m/s. Moreover, since the different delays that face any packet at each node due to processing were not accounted for neither in L-B nor in B-ERS; we assume that each packet in the original on-demand routing algorithm faces delay of one unit of time at each node. This is to utilise the multiplicative identity so $NTT\ =\ 1$. Furthermore, the hop count is assumed to be the network size divided by 10. Networks are of sizes 20, 30… 100 nodes so hop counts are 2, 3 … 10 for different sources and route finders under the same environment. TLRDA-C inherits the same values for T_{Class1} and T_{Class2} from TLRDA-D, introduced in Chapter 4, where the delay equals $2LP_r$.

Case 1:

For the first possibility (Case 1 where $h_s(f) \leq \frac{LP_r}{2}$) when the route request is within neighbourhood region at time t; we conduct a comparison between TLRDA-C and both B-ERS

and L-B. Such comparison aims at studying the behaviour of these algorithms and evaluates the growth of RDT and RRL using various values for the hop count as shown in Figure 5-7 and Figure 5-8 respectively. In TLRDA-C, LP_r should satisfy the condition of Case 1 where $2h_s(f) \leq LP_r$. LP_r was given a range of values depending on the value of $h_s(f)$ where $LP_r = 2h_s(f)$, $LP_r = 2h_s(f) + 1$, and $LP_r = 2h_s(f) + 2$ for each hop count ignoring other values as LP_r has a finite value and does not grow far from $h_s(f)$.

Figure 5-7 shows the route discovery time for all three algorithms TLRDA-C, L-B, and B-ERS. The results reveal that TLRDA-C is the quickest among the three algorithms because TLRDA-C does not delay route requests in their neighbourhood region also delaying route requests in their beyond-neighbourhood region reduces the congestion as explained before when analysing TLRDA-D in Chapter 4. On the other hand, B-ERS introduces a delay equal double the hop count from the start while L-B always slowdown both route requests and route replies.

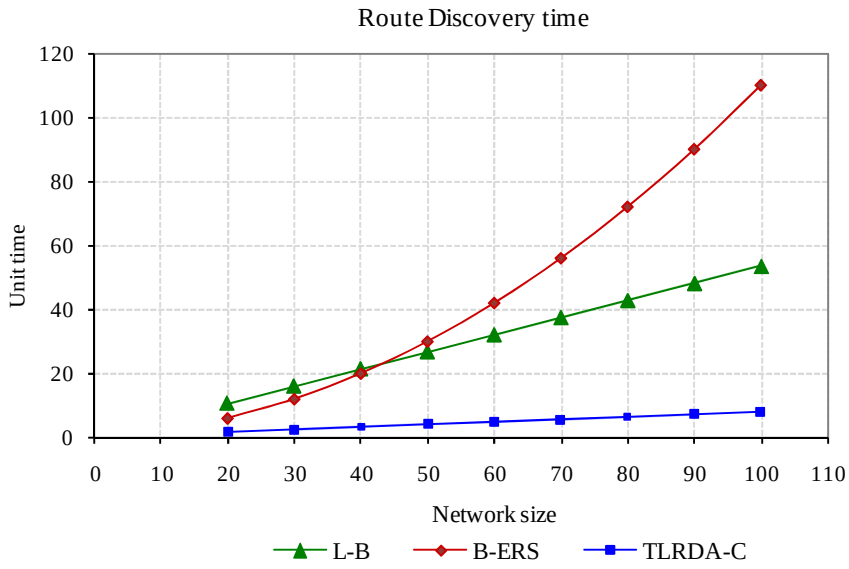

Figure 5-7: Route discovery time versus network size when Case 1 is true.

For the route request lifetime (RRL), we conduct a comparison among TLRDA-C, L-B, and B-ERS. Such a comparison aims at showing the behaviour of our algorithm and evaluating the growth of RRL. The values for RRL at each hop count were averaged to produce TLRDA-C line graph. Figure 5-8 depicts the performance of TLRDA-C compared to both L-B and B-ERS using different hop counts and shows that TLRDA-C reduces RRL from 83% to 84% over L-B and from 77% to 91% over B-ERS. The route request lifetime for TLRDA-C is lower which means that

TLRDA-C discards fulfilled route requests quicker. Since TLRDA-C broadcasts unanswered route requests quicker than both B-ERS and L-B, chase packets are initiated earlier in TLRDA-C so the catching can happen earlier which reduces the route request lifetime. B-ERS introduces a delay at each intermediate node and this delay increases with the increment of hop count for the same route request as clearly shown in Figure 5-8.

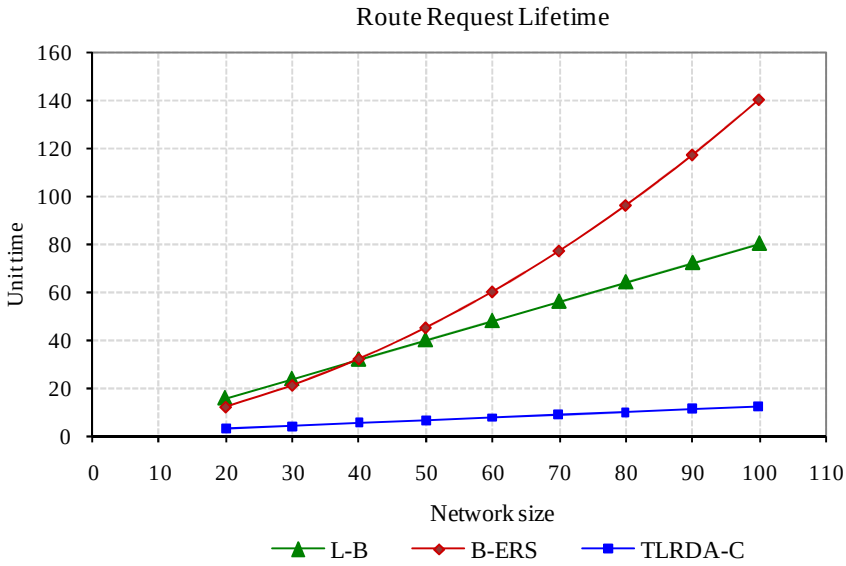

Figure 5-8: Route request lifetime versus network size when Case 1 is true.

Comparing the route discovery time for all three algorithms as in Figure 5-7 and the route request lifetime shown in Figure 5-8, the superiority of TLRDA-C can clearly be seen for both times RDT and RRL, i.e. less latency.

Case 2:

For the second possibility (Case 2 where $h_s(f) > \frac{LP_r}{2}$) when the route request is within the beyond-neighbourhood region at time t; we conducted a comparison also between TLRDA-C and both B-ERS and L-B. The results are depicted in Figure 5-9 and Figure 5-10. Values for the hop count were varied from 2 to 10 incremented by 1. LP_r has a finite range of values that satisfy the condition of Case 2 ($2h_s(f) > LP_r$). So LP_r was given all integer values from 1 to $2h_s(f) - 1$ then the values for RDT and RRL were averaged for each metric at each hop count to produce the graphs for TLRDA-C in Figure 5-9 and Figure 5-10 respectively.

Figure 5-9 shows that TLRDA-C discovers new routes quicker than both L-B and B-ERS.

TLRDA-C improves *RDT* up to 69% over L-B while the improvement ranges from 45% to 72% over B-ERS. This is due to the fast propagation of the route request within its neighbourhood region.

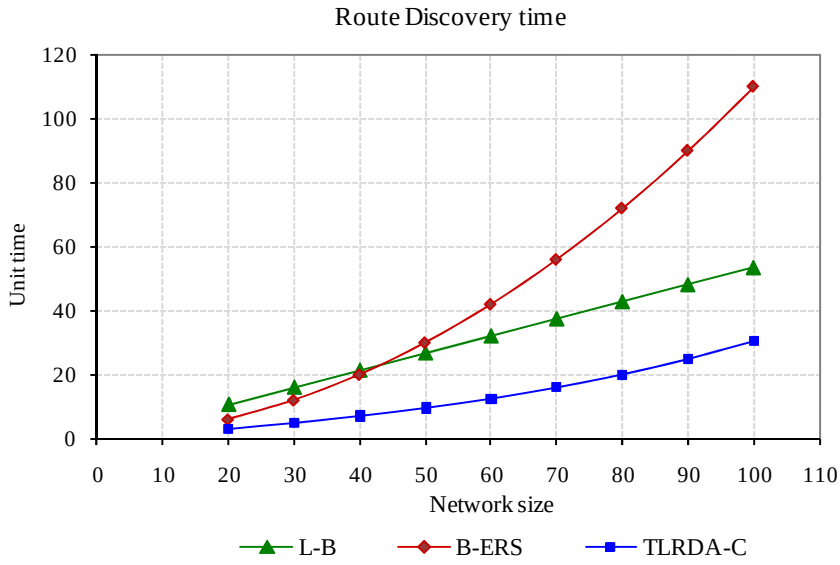

Figure 5-9: Route discovery time versus network size when Case 2 is true.

For route request lifetime (RRL), we conduct a comparison among all three algorithms. Figure 5-10 shows RRL for all three algorithms. The improvement in *RRL* in favour of TLRDA-C compared to both L-B and B-ERS using different hop count values can be clearly seen in this figure. TLRDA-C reduces the total broadcast time for the route request. This reduction ranges from 25% to 33% over L-B and up to 57% over B-ERS. TLRDA-C and L-B relates chase time to *RRL* so the success of the catching process is highly likely to happen.

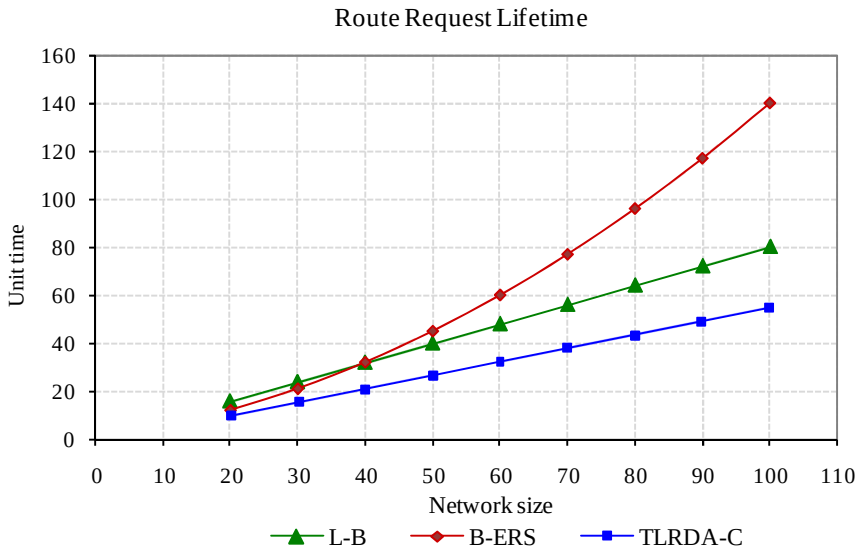

Figure 5-10: Route request lifetime versus network size when Case 2 is true.

Comparing the results of RDT and RRL for TLRDA-C against those of the L-B and B-ERS algorithms shows the superiority of TLRDA-C resulting in lower latency in both Case 1 and Case 2.

5.4 Simulation analysis

The amount and time of adding a delay to route request propagation are essential. If the amount of delay is large, the catching process will be quicker. Also, if this delay is imposed before discovering the required route, both actual data and chase packets will be delayed as well and will even be more expensive in terms of latency.

Simulation has been conducted to experiment with our algorithm, TLRDA-C, against simple flooding used in AODV, B-ERS, and L-B algorithms using ns2 simulator version 2.29 [41]. TLRDA-C, L-B, and B-ERS were implemented as modifications to the existing AODV implementation. B-ERS and L-B use the same chase packet format as in TLRDA-C. Moreover, TLRDA-C adds a byte to the route request packet to store the value of LP_r while B-ERS adds one byte to the route reply packet to carry the value of $h_s(f)$ to the source node. In AODV, NTT is equal to 40ms and is the same for all the algorithms. The amount of delay added to B-ERS and TLRDA-C are specified in Table 5-2.

Table 5-2: Amount of added delay in the B-ERS and TLRDA-C algorithms.

<table>
<tr><th colspan="5">Amount of added delay</th></tr>
<tr><td></td><td colspan="2">Route request</td><td>Route reply</td><td>Chase packet</td></tr>
<tr><td>Blocking-ERS (B-ERS)</td><td colspan="2">$2hop\_count * NTT$</td><td>none</td><td>none</td></tr>
<tr><td rowspan="2">TLRDA-C</td><td>Within neighbourhood</td><td>Beyond-neighbourhood</td><td rowspan="2">none</td><td rowspan="2">none</td></tr>
<tr><td>none</td><td>$2LP * NTT$</td></tr>
</table>

Table 5-3 shows the time slots used in L-B in the presence of route request and route reply or chase packet ready for transmission where the timer (t) is initialised to zero and is reset whenever its value reaches NTT value.

Table 5-3: Transmission slots used in the L-B algorithm.

Limited Broadcasting (L-B)		
	Route request	Route reply or chase packet
Transmission slot	$0 \leq t < (NTT/4)$	$(NTT/4) \leq t < NTT$

In the rest of this section, the comparison metrics include end-to-end delay, the average route request latency, routing overhead, and packet loss. The average route request latency and end-to-end delay are used to study the network latency while the overhead is studied by routing overhead and congestion is studied by packet loss. A new metric is used in this chapter to measure the success rate of the catching process by utilising network coverage. The network coverage is measured as the number of receiving nodes per route request where a node is counted as one if it received one or more copies of the same route request. This metric provides an indication of the success rate of the chasing mechanism where each algorithm is compared to AODV because it gives complete coverage when simple flooding is used.

The simulation analysis considers the effects of network size, traffic load, and mobility as stated earlier in the second chapter as in Table 2-3.

5.4.1 Effect of network size

Figures 5-11 to 5-16 display the performance results of comparing TLRDA-C against AODV, B-ERS, and L-B algorithms using networks with different sizes. The number of nodes is multiple of 10 starting from 20 until 100 with a minimum speed of 1m/s and a maximum speed of 15m/s. The number of communication sessions is ten.

Success rate:

Figure 5-11 shows that TLRDA-C achieves a better success rate for the catching process than the other algorithms: AODV, B-ERS, and L-B. The rate of success for the catching process is determined by the amount of coverage. The optimal success rate is when the coverage equals to $h_s(f)$ but this cannot be obtained efficiently without the use of external resources. When the network is covered completely by a route request, while the algorithm uses the chasing technique, the rate of the success in the chasing process is zero; i.e. less coverage means higher success rate. In AODV, where simple flooding is used, there are no chase packets so the network is almost covered by default where the coverage is 100% most of the time. In B-ERS, the coverage is nearly equal to AODV because the discard of the chase packet before catching the associated route request makes the fulfilled route requests cover the whole network most of the time. B-ERS's coverage is almost the same as that of AODV with a little improvement. This improvement might be due to low catching or packet loss especially when contention is high as in moderate size networks. L-B succeeds in the catching process to some extent and its coverage is less than that of AODV by 55% in small size and 37% in moderate size network due to the small amount of delay added to route requests compared to B-ERS and TLRDA-C which makes the fulfilled route requests propagates further in the network. TLRDA-C achieves the best success rate among all the four algorithms. Its coverage is less by 69% in small size network and by 85% in moderate size network compared to AODV and less than B-ERS by 67% to 85% while it is less by 31% to 76% compared to L-B coverage.

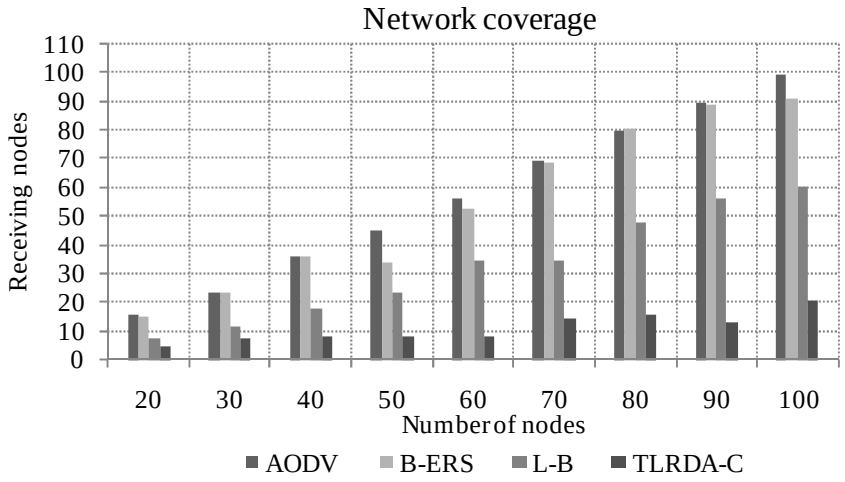

Figure 5-11: Network coverage versus network size with 10 communication sessions and 15m/s as maximum speed.

Latency:

Figure 5-12 explores the end-to-end delay for TLRDA-C, L-B, B-ERS, and AODV. The end-to-end delay increases with the network size for all four algorithms because when the network size increases the hop count of a path increases which in turn increases the discovery time. TLRDA-C inherits the positive features from TLRDA-D such as the low average of route discovery time which reduces the end-to-end delay because the discovery time is included in the end-to-end delay. Thus, it reduces the average end-to-end delay more than L-B, B-ERS, and AODV.

TLRDA-C achieves a lower end-to-end delay due to the faster propagation of the route request within its neighbourhood region remembering that TLRDA-C broadcast with less contention as in TLRDA-D. The reason behind the increment in the average end-to-end delay in both B-ERS and L-B is the delaying of route requests from start and before discovering the required route. TLRDA-C's improvement of the average end-to-end delay ranges from 58% to 67% over AODV, 62% to 70% over B-ERS, and 51% to 68% over L-B. If the route discovery process is fast, the reply will reach the source node earlier which gives the source node the opportunity to broadcast the chase packet and the application data earlier. Application data is stored within the source node until a valid route is found. This affects the end-to-end delay so if the discovery is a quick process, the data are stored for less time which reduces the end-to-end delay.

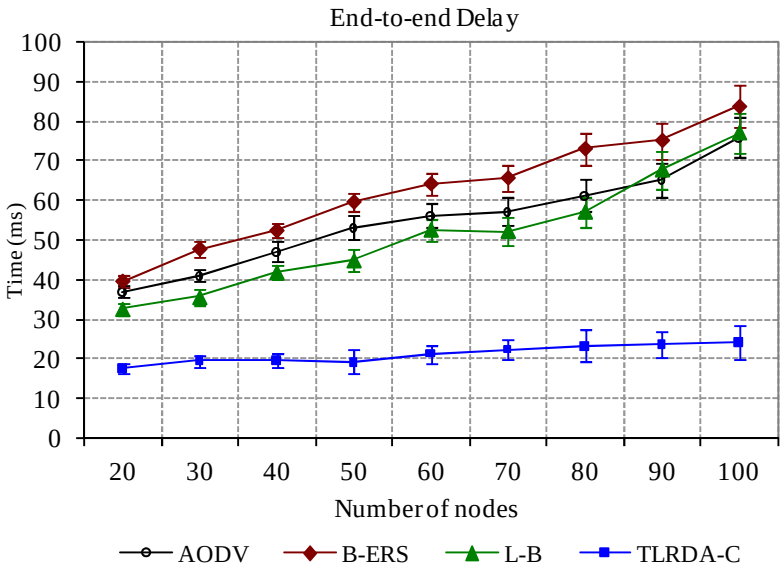

Figure 5-12: End-to-end delay versus network size with 10 communication sessions and 15m/s as maximum speed.

Figure 5-13 shows the superiority of TLRDA-C in minimising the average of route request latency. The average route request latency of TLRDA-C is reduced due to the better success rate in the catching process for TLRDA-C as shown above in Figure 5-11. The route requests in B-ERS reside in the network more than AODV; the reasons behind this phenomenon are: i) the large delay always added to route requests, ii) low success rate of catching fulfilled route requests. TLRDA-C improves the average of route request latency by 46% to 57% over AODV, 64% to 83% over B-ERS, and 35% to 50% over L-B.

TLRDA-C sends the chase packet earlier without adding any extra delay to the chase packet propagation which makes the chasing process quicker than L-B. Since TLRDA-C achieves the lowest end-to-end delay among all four algorithms, the chase packets starts earlier in TLRDA-C which give chase packets a better opportunity to succeed.

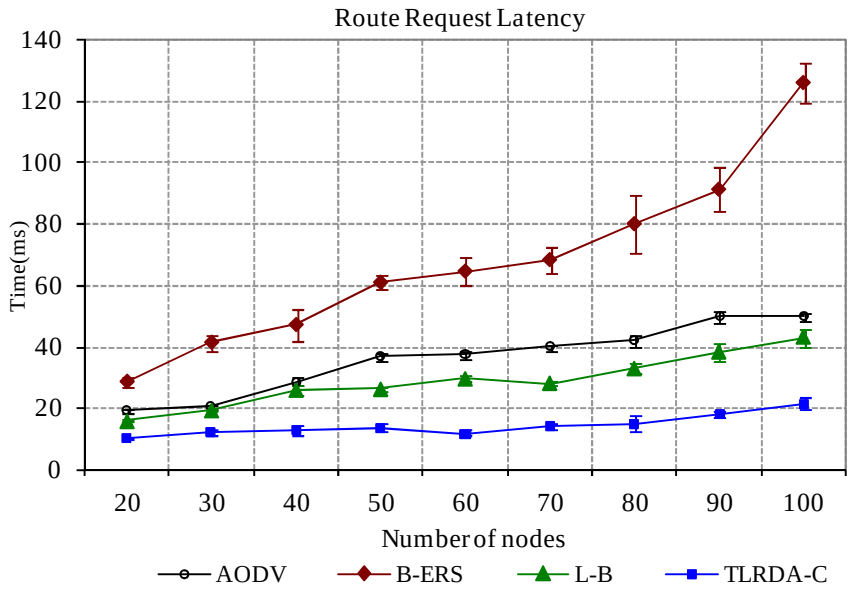

Figure 5-13: Average route request latency versus network size with 10 communication sessions and 15m/s as maximum speed.

TLRDA-C exhibits a better performance with respect to the route request latency and route discovery time because it adds delay in the beyond-neighbourhood region only to minimise delay in the route discovery process and chase packets broadcast until catching their associated route request packets; resulting in a better chance of successful catch and reduces the average of route request latency. Moreover, delaying then stopping the fulfilled route requests reduces network congestion which improves network latency in terms of reducing the end-to-end delay and route request latency.

Overhead:

The route request overhead metric is the number of route requests received in the whole network. We are interested in knowing the routing overhead before and after adding the number of chase packets received. The route request overhead in TLRDA-C is lower than that of AODV, B-ERS, and L-B as shown in Figure 5-14. The overhead increases with the increment of network density regardless of the algorithm used because the average number of route requests received might increase more when the route request spreads deeper in the network as it is broadcast in all direction and may reach more nodes each time it propagates further. The success of the catching process frees the network from more fulfilled route requests which improves both network latency and the overhead. This figure highlights the fact that the difference in route request overhead between TLRDA-C and the other algorithms increases with moderate network. TLRDA-C

improves the average number of route request received by 59% to 71% over AODV, 63% to 78% over B-ERS, 38% to 64% over L-B.

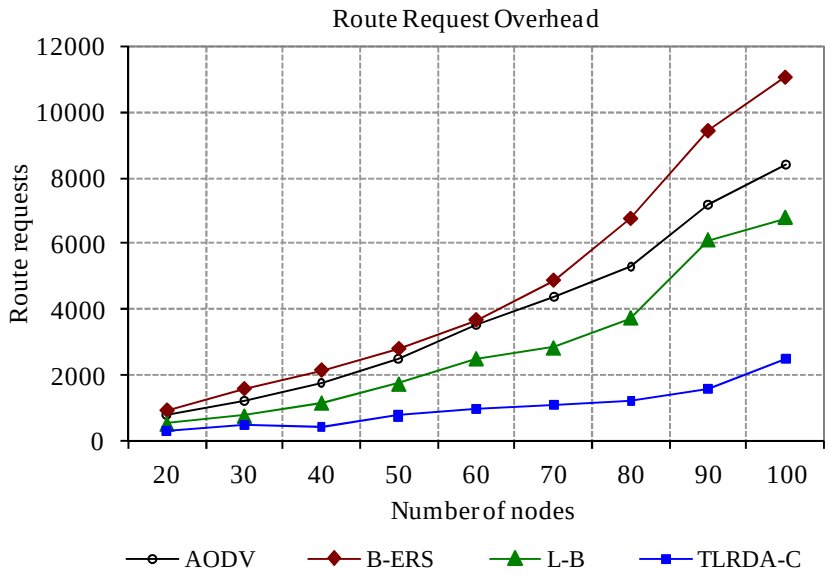

Figure 5-14: Route requests overhead versus network size with 10 communication sessions and 15m/s as maximum speed.

Moreover, Figure 5-15 shows the routing overhead for all four algorithms where TLRDA-C reduces routing overhead more than AODV, B-ERS, and L-B. This improvement increases in moderate network which intern improves power consumption and gives better bandwidth utilisation. TLRDA-C's improvement of the routing overhead is 17% to 51% over AODV, 32% to 63% over B-ERS, and 38% to 72% over L-B. The number of received chase packets in B-ERS is less than TLRDA-C by up to 20% but because of higher number of received route requests in B-ERS, routing overhead is lower in TLRDA-C.

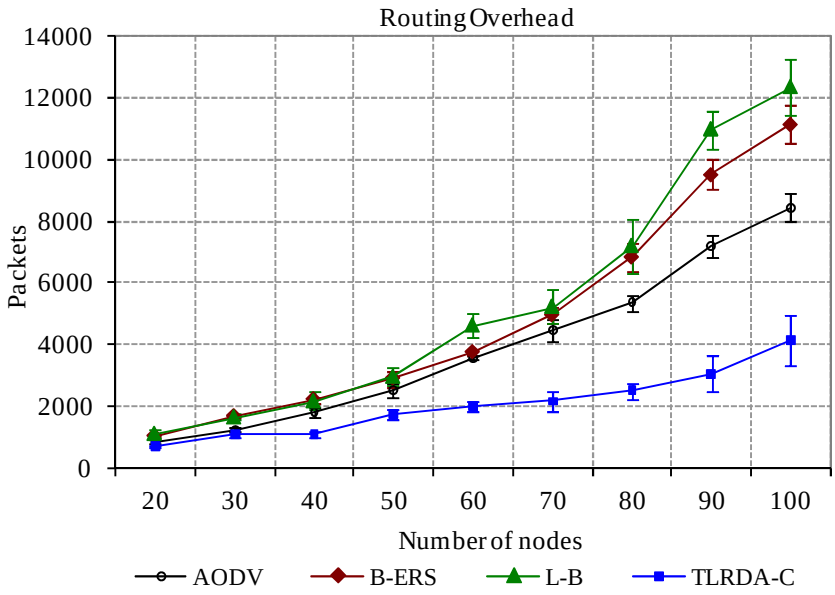

Figure 5-15: Routing overhead versus network size with 10 communication sessions and 15m/s as maximum speed.

Congestion:

TLRDA-C reduces packet loss in the whole network compared to AODV, B-ERS, and L-B as shown below in Figure 5-16 because the network in TLRDA-C is less congested, as in TLRDA-D, which saves more packets especially with moderate size network environment. TLRDA-C improves packet loss by 21% to 59% over AODV, and up to 68% over B-ERS, and 22% to 75% over L-B. Simple flooding is very costly in moderate size networks in terms of overhead because increasing number of nodes will increase the number of hops for any single packet. This increases the channel contention and congests the network leading to increment in packet loss. However, in TLRDA-C the success of freeing the network from unwanted route requests saves more important packets from being dropped while needed. Therefore, the network performance is improved for TLRDA-C by reducing latency and overhead due to the higher success rate of the catching process. So the quick broadcasting in a less congested environment such as TLRDA-C improves the network performance in terms of latency, overhead, and congestion level. This improvement increases with moderate networks.

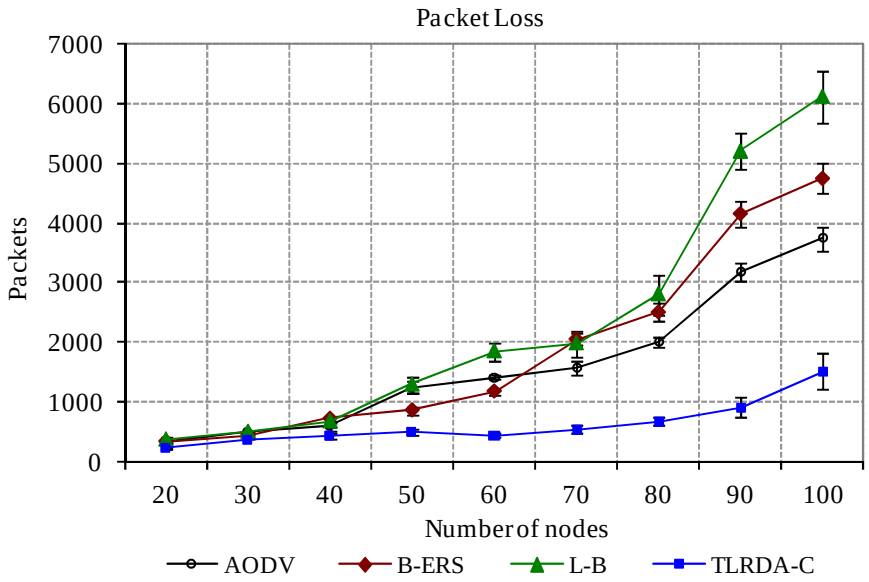

Figure 5-16: Packet loss versus network size with 10 communication sessions and 15m/s as maximum speed.

5.4.2 Effect of traffic load

Figures 5-17 to 5-21 display the results of running our algorithm, TLRDA-C, against AODV, B-ERS, and L-B using networks of size 70 nodes with a random speed ranging between 1m/s and 15m/s. The amount of traffic ranges from 5 to 35 communication sessions incremented by five.

Success rate:

Figure 5-17 demonstrates how much the network is covered. From this figure we can compare the success rate of the chasing technique in stopping the fulfilled route requests in all of the algorithms that use chase technique, TLRDA-C, B-ERS, and L-B. AODV covers the network completely as expected from simple flooding but when the network is injected with heavy traffic as in 35 communication sessions the number of receiving nodes was almost double the network size which means that some of the route requests were reinitiated more than once by the source node due to the high congestion and contention. At 30 communication sessions, B-ERS succeeded in some of the chasing process but still its success rate is lower than both TLRDA-C and L-B. TLRDA-C has the best success rate among all four algorithms. TLRDA-C's coverage is less by 90% in light traffic and 94% in heavy traffic compared to AODV and less than B-ERS by 83% to 84% while it is less by 53% to 60% compared to L-B coverage. So the success rate of TLRDA-C improves more with heavy traffic load.

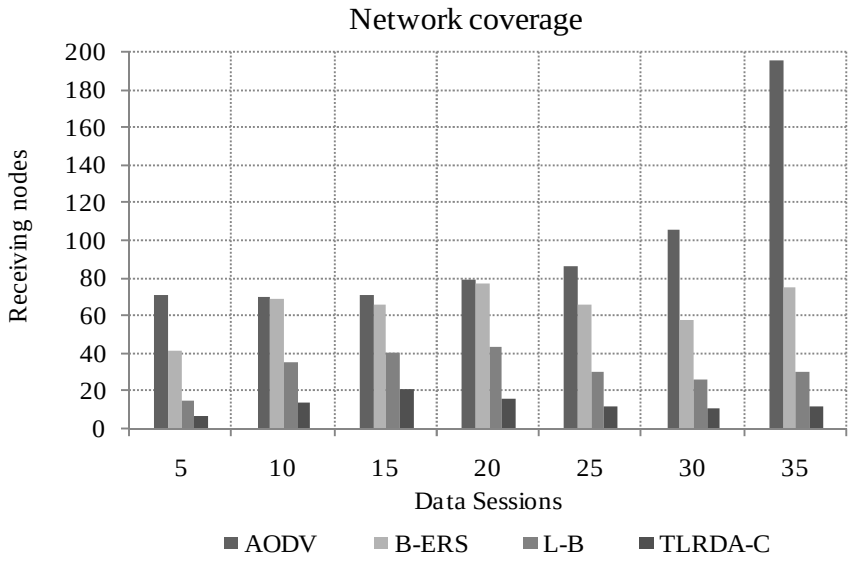

Figure 5-17: Network coverage versus traffic load in networks of 70 nodes and 15m/s as maximum speed.

Latency:

Figure 5-18 shows that TLRDA-C improves the end-to-end delay over AODV, B-ERS, and L-B. This improvement due to the quicker broadcasting while the required route is not discovered yet compared to B-ERS and L-B. TLRDA-C works in a less congested environment compared to AODV.

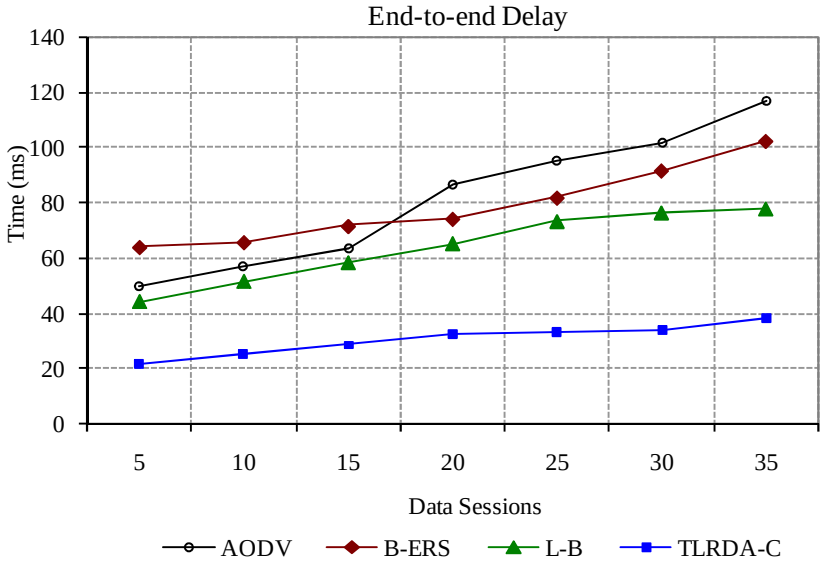

Figure 5-18: End-to-end delay versus traffic load in networks of 70 nodes and 15m/s as maximum speed.

B-ERS has higher end-to-end delay compared to AODV in light traffic networks because B-ERS introduces the delay from the start and before discovering the needed route. However, in heavy

traffic networks B-ERS reduces the contention level more than AODV reducing end-to-end delay.

TLRDA-C's improvement ranges from 55% to 65% over AODV, 59% to 63% over B-ERS, and 51% to 56% over L-B. When the traffic load increases, channel contention increases which increases the end-to-end delay in all four algorithms.

Figure 5-19 reveals the superiority of TLRDA-C among the four algorithms in terms of the average of route request latency because it achieves higher success rate in the catching process and avoid delaying route request before discovering the required route. The route request latency increases when the traffic load increases due to the increment of the number of packets in the network. This adds more contention and may result in more collisions. TLRDA-C improves the average of route request latency by 53% to 67% over AODV, 67% and 72% over B-ERS, and 36% to 50% over L-B.

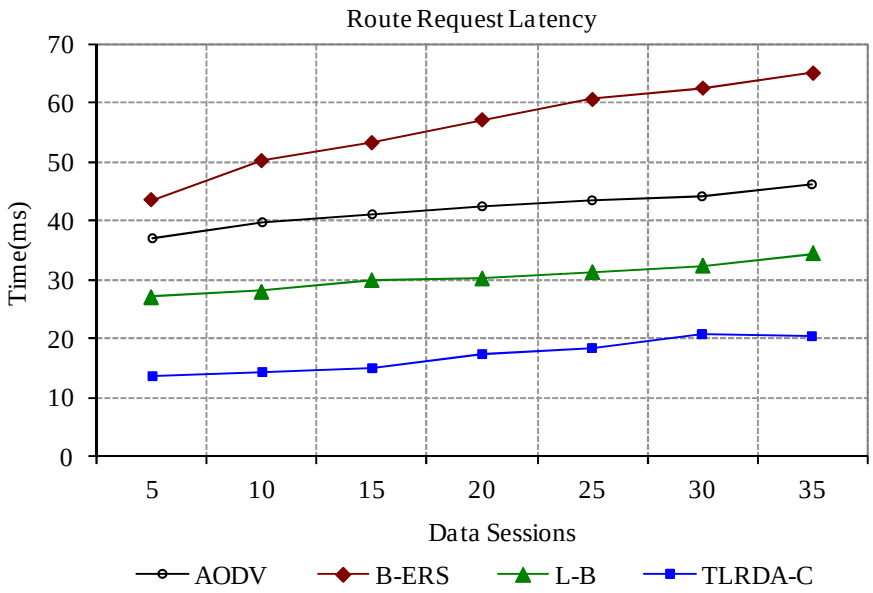

Figure 5-19: Route request latency versus traffic load in networks of 70 nodes and 15m/s as maximum speed.

Overhead:

Figure 5-20 expresses the routing overhead for all four algorithms and shows that TLRDA-C achieves lower routing overhead than AODV, B-ERS, and L-B. Such improvement increases with the increment of traffic load which improves both power consumption and bandwidth utilisation. The improvement of the routing overhead in TLRDA-C is 51% to 81% over AODV, up to 60%

over B-ERS, and 55% to 61% over L-B.

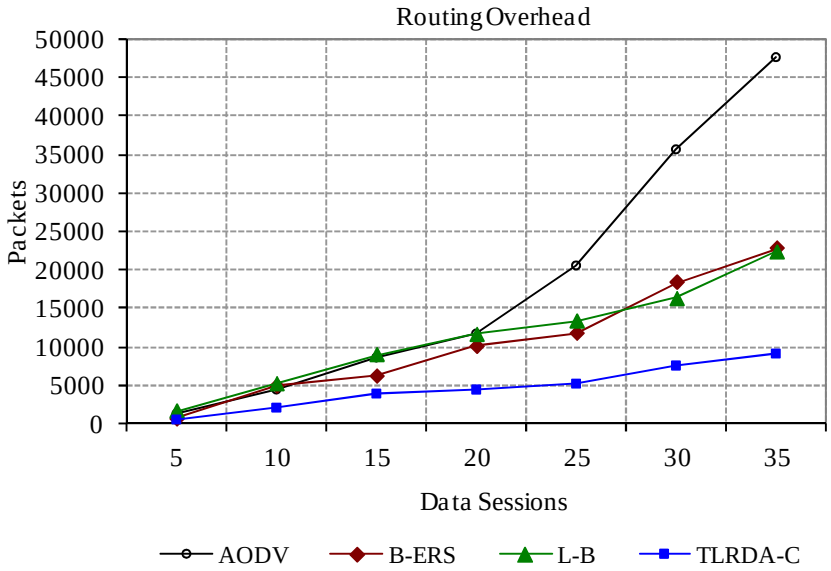

Figure 5-20: Routing overhead versus traffic load in networks of 70 nodes and 15m/s as maximum speed.

Congestion:

TLRDA-C incurs less packet loss in the whole network compared to AODV, B-ERS, and L-B as shown below in Figure 5-21 because the network in TLRDA-C is less congested. The packet loss is increased with the increment of traffic load for all four algorithms. However, TLRDA-C improves packet loss by 32% to 67% over AODV, 22% to 68% over B-ERS, and 40% to 80% over L-B.

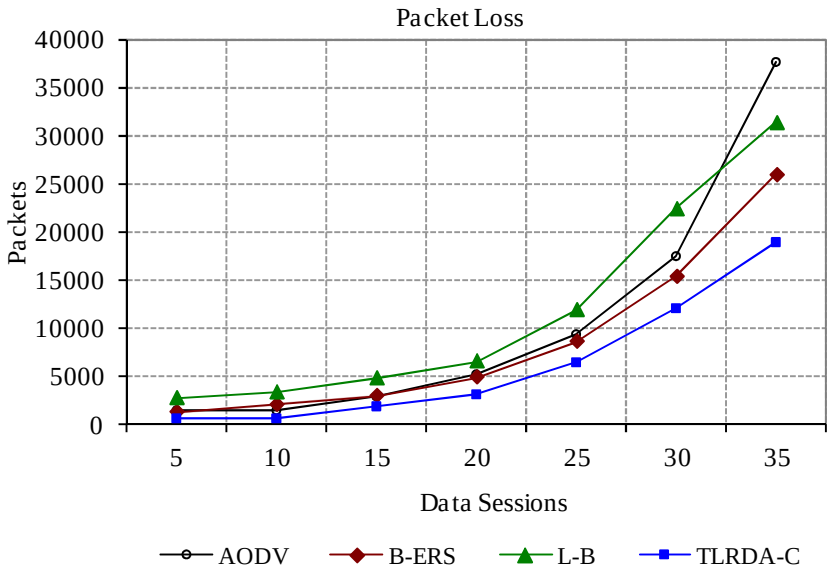

Figure 5-21: Packet Loss versus traffic load in networks of 70 nodes and 15m/s as maximum speed.

5.4.3 Effect of mobility

Figures 5-22 to 5-26 were extracted from simulating the four algorithms using networks of size 70 nodes. These networks use six different maximum speeds where the actual speed is randomly selected from [1, maximum speed]. The six maximum speeds take the following values: 2, 5, 7, 10, 13, and 15m/s respectively. The communication sessions was fixed to be 10.

Success rate:

Figure 5-22 demonstrates network coverage as an indicator of the success rate of the catching process. AODV covers the network almost completely especially with fast networks because of the simple flooding. TLRDA-C has the best success rate among the four algorithms. TLRDA-C's coverage is less by 79% in slow networks and 86% in fast networks compared to AODV and less than B-ERS by 79% to 85% while it is less by 56% to 69% compared to L-B coverage.

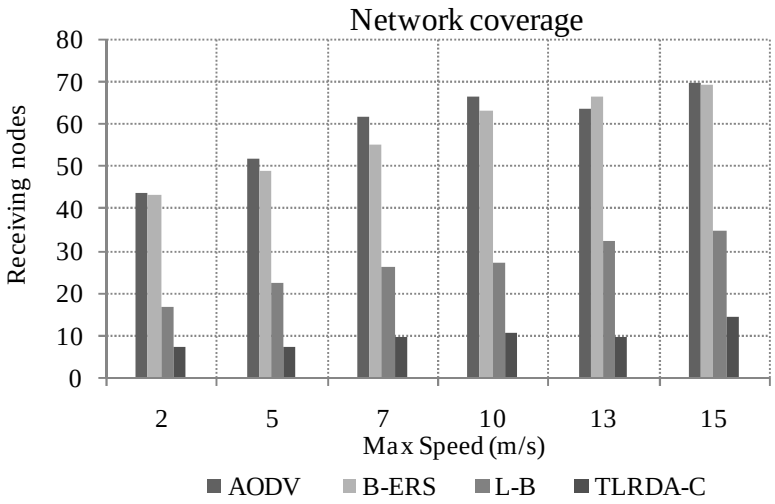

Figure 5-22: Network coverage versus maximum speed in networks of 70 nodes and 10 communication sessions.

Latency:

The average of end-to-end delay increases with fast networks regardless of the algorithm used. However, TLRDA-C offers better end-to-end delay over AODV, B-ERS, and L-B as shown in Figure 5-23. This improvement is due to less congested environment among the four algorithms and/or quick broadcasting within the neighbourhood region compared to B-ERS and L-B. TLRDA-C improves route request latency by 54% to 61% over AODV, 63% to 66% over B-ERS, and 41% to 52% over L-B.

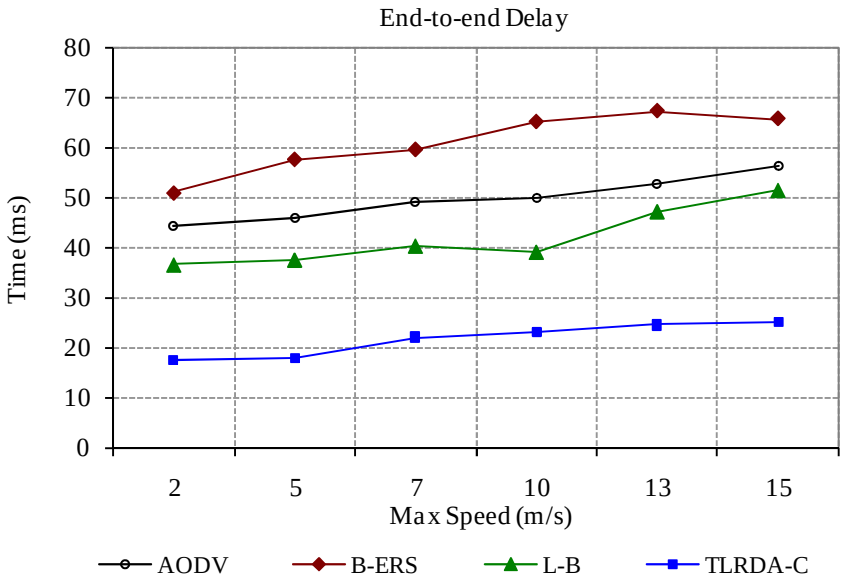

Figure 5-23: End-to-end delay versus maximum speed in networks of 70 nodes and 10 communication sessions.

Figure 5-24 shows a great reduction in route requests latency for TLRDA-C over B-ERS, L-B, and AODV regardless of speed which improves the network performance. As mentioned earlier, this improvement is due to the higher success rate of TLRDA-C in the catching process. The route request latency increases slightly with the increment of speed due to link breakage regardless of the algorithm used. However, TLRDA-C improves the average of route request latency by 64% to 71% over AODV, 72% to 77% over B-ERS, and 49% to 62% over L-B.

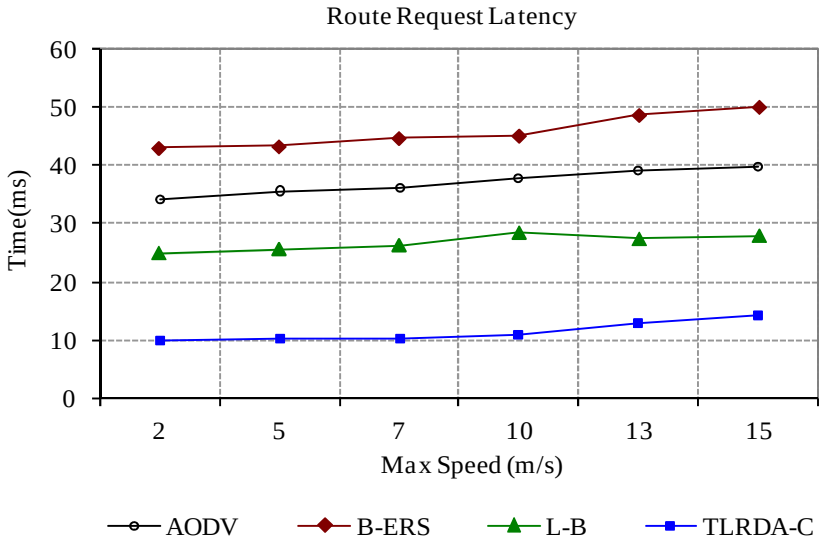

Figure 5-24: Route request latency versus maximum speed in networks of 70 nodes and 10 communication sessions.

Overhead:

Routing overhead increases with fast networks with all the experimented algorithms. L-B incurs higher routing overhead due to the high number of route requests and the high number of chase packets rebroadcast through the network. However, TLRDA-C incurs lower routing overhead than AODV, B-ERS, and L-B as shown in Figure 5-25 which should improve both power consumption and bandwidth utilisation as mentioned before. The improvement of the routing overhead in TLRDA-C ranges from 49% to 62% over AODV, 57% to 67% over B-ERS, and 56% to 64% over L-B.

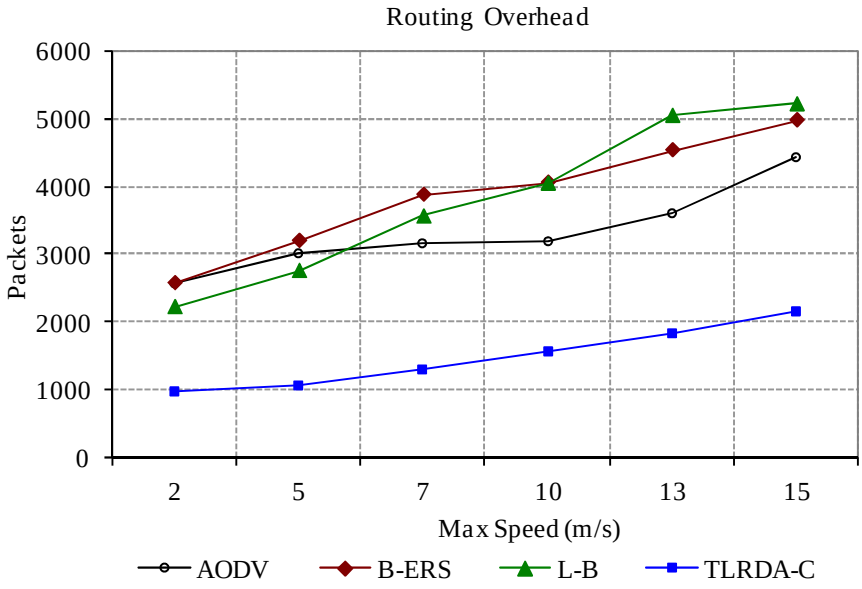

Figure 5-25: Routing overhead versus maximum speed in networks of 70 nodes and 10 communication sessions.

Congestion:

TLRDA-C loses fewer packets compared to AODV, B-ERS, and L-B as shown below in Figure 5-26 because the network is less congested as in TLRDA-D. The packet loss is increased with the increment of maximum speed for all four algorithms. However, TLRDA-C improves packet loss by 63% to 78% over AODV, 61% to 79% over B-ERS, and 63% to 82% over L-B.

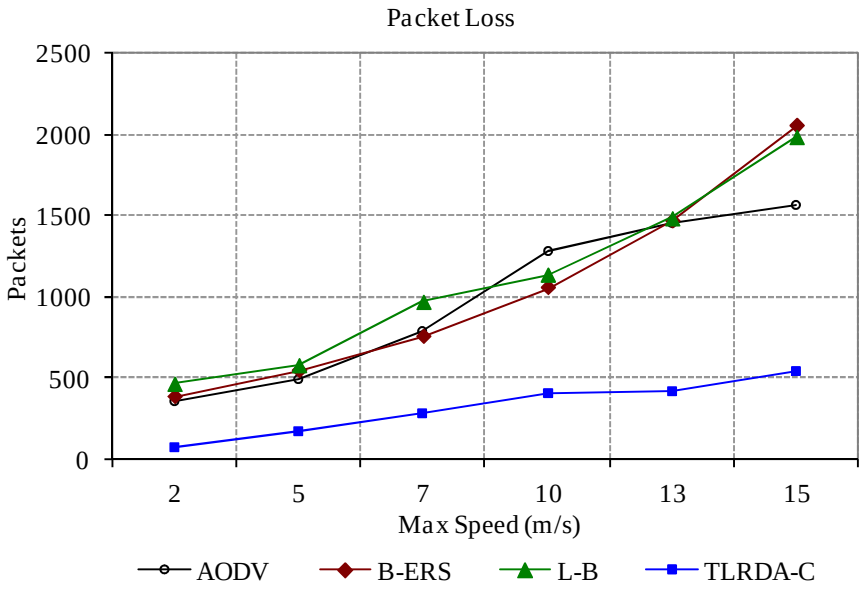

Figure 5-26: Packet loss versus maximum speed in networks of 70 nodes and 10 communication sessions.

Looking at the three-performance analyses, our algorithm outperforms AODV, B-ERS, and L-B regardless of network size, traffic load, or speed. TLRDA-C reduces route request latency and end-to-end delay also reduces the routing overhead and packet loss so TLRDA-C reduces network latency and overhead.

5.5 Summary of simulation results

Simulation experiments have been conducted using the same simulation parameters as in the previous sections to study the performance of TLRDA-C and compare its performance with that of AODV, B-ERS, and L-B from network size, traffic load, or mobility prospective where the same trends have been observed when comparing the performance of TLRDA-C with the performance of AODV, B-ERS, and L-B. The percentages of TLRDA-C improvements are summarised in Table 5-4 for the following metrics: end-to-end delay, route request latency, and routing overhead as well as packet loss which were also shown in Figure 5-11 to Figure 5-26 earlier. Furthermore for simplicity, Table 5-4 shows the result of the network size analysis for small and moderate size networks, traffic load analysis for light and heavy traffic, and the mobility analysis for slow and fast networks.

Table 5-4: Summary of the improvements for TLRDA-C over AODV, B-ERS, and L-B.

Cases	Algorithm	Route Request Latency		End-to-end delay		Routing Overhead		Packet Loss	
Network size		Small	Moderate	Small	Moderate	Small	Moderate	Small	Moderate
	AODV	46%	57%	53%	68%	17%	51%	28%	59%
	L-B	35%	50%	47%	68%	38%	66%	33%	75%
	B-ERS	64%	83%	56%	71%	32%	63%	25%	68%
Traffic load		Light	Heavy	Light	Heavy	Light	Heavy	Light	Heavy
	AODV	63%	56%	56%	67%	51%	81%	67%	50%
	L-B	50%	41%	51%	51%	59%	59%	79%	40%
	B-ERS	69%	69%	66%	63%	1%	60%	57%	27%
Mobility		Slow	Fast	Slow	Fast	Slow	Fast	Slow	Fast
	AODV	61%	64%	61%	56%	62%	51%	78%	65%
	L-B	60%	49%	52%	52%	56%	59%	82%	72%
	B-ERS	77%	72%	66%	62%	62%	57%	79%	73%

TLRDA-C outperforms all the three algorithms by reducing end-to-end delay due to the reduction in network congestion, as in TLRDA-D. TLRDA-C also improves route request latency, routing overhead, packet loss due to the higher success rate of the catching process as shown in the network coverage metrics.

5.6 Comparison between TLRDA-C and TLRDA-D

In Chapter 4, TLRDA-D was introduced and its performance was studied. Nodes in TLRDA-D broadcast route requests within their source node's neighbourhood region according to the routing algorithm used. Afterwards, nodes broadcast the route request with a delay equal to $2LP_r * NTT$ outside such neighbourhood.

The data extracted from the simulation runs show that TLRDA-D succeeds in improving the discovery time which improves the end-to-end delay but with a cost of higher average route request latency and overhead. This extra overhead is wasting network resources i.e. power and bandwidth which affect the network performance. TLRDA-C has been introduced in this Chapter to overcome such deficiency in TLRDA-D i.e. aiming at reducing route request latency and improving the routing overhead. The effect of network size for TLRDA-D and TLRDA-C are compared to measure the success of TLRDA-C in reducing latency and overhead over TLRDA-D.

Latency:

TLRDA-D and TLRDA-C improve network congestion and reduce channel contention; so both of them have almost the same end-to-end delay and packet loss where the difference is negligible. Figure 5-27 shows the success of TLRDA-C in minimising the average of route request latency. The average route request latency of TLRDA-C is reduced because the catching process discards unwanted route requests in the beyond-neighbourhood region which has average route request latency larger than the route request within neighbourhood region due to the added delay. TLRDA-C improves the average of route request latency by 44% to 89% over TLRDA-D.

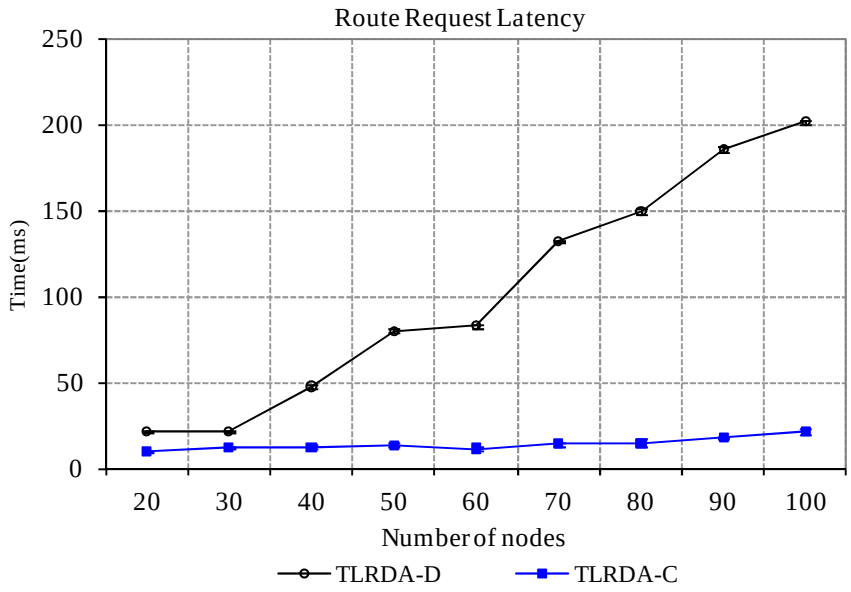

Figure 5-27: Average route request latency versus network size with 10 communication sessions and 15m/s as maximum speed.

Overhead:

TLRDA-C incurs a lower routing overhead than TLRDA-D as shown in Figure 5-28. The overhead increases with the increase in the network size in both algorithms because the average number of route request received increases with the increment of network size in a fixed arena. The success of the catching process in TLRDA-C frees fulfilled route requests thus improving both network latency and reducing overhead. Moreover, TLRDA-C improves the average routing overhead up to 57% over TLRDA-D. Both algorithms experience almost the same packet loss due to their ability to reduce network congestion.

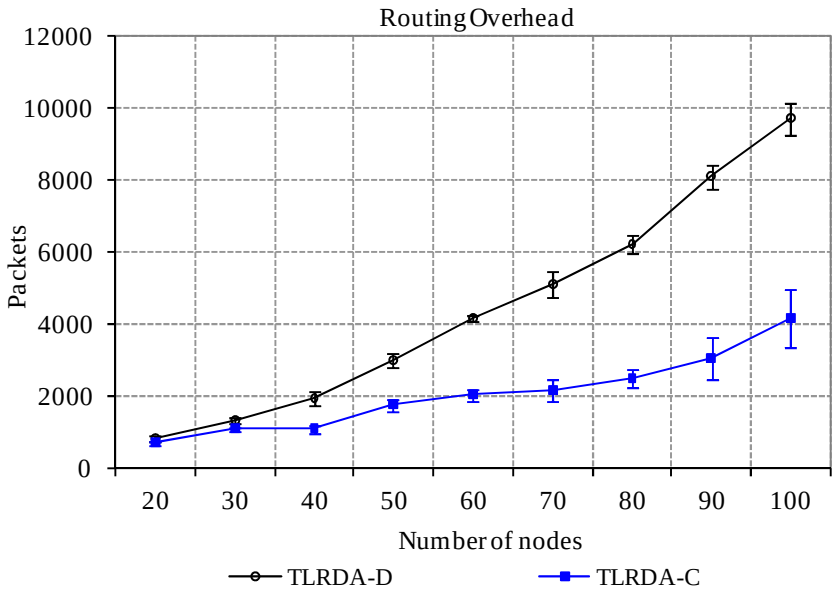

Figure 5-28: Routing overhead versus network size with 10 communication sessions and 15m/s as maximum speed.

The traffic load and mobility analyses show the same behaviour as the network size. The results of the comparison are summarised in Table 5-5.

Table 5-5: Summary of the improvements for TLRDA-C over TLRDA-D.

Cases	Route Request Latency		Routing Overhead	
Network size	Small	Moderate	Small	Moderate
	52%	89%	17%	57%
Traffic load	Light	Heavy	Light	Heavy
	87%	88%	67%	83%
Mobility	Slow	Fast	Slow	Fast
	92%	89%	66%	58%

In summary, the comparison between TLRDA-C and TLRDA-D demonstrates the success of TLRDA-C in reducing route request latency and routing overhead which were the deficiency in TLRDA-D for improving the discovery process.

5.7 Conclusions

In this chapter, a new traffic locality oriented route discovery algorithm, referred to as TLRDA-C, was developed for MANETs applications that exhibit traffic locality. TLRDA-C is an improvement over TLRDA-D to reduce route request latency and overhead by utilising chase

packets. TLRDA-C works by establishing a neighbourhood region for each active source node that includes most of the destinations and broadcasts the route requests with no extra delay within such region to improve the end-to-end delay. In order to provide a better chance for the chase packets to catch their associated route requests, the algorithm delays the propagation of the route requests in the beyond-neighbourhood region which in turn helps to minimise the network congestion. The algorithm continuously updates the neighbourhood boundary to provide a better performance.

A detailed performance evaluation using mathematical modelling and simulation for our new algorithm, TLRDA-C, was provided and compared against existing algorithms. Our simulation analysis has shown that TLRDA-C has lower route request latency, lower end-to-end delay, less routing overhead, and fewer lost packets compared to AODV, Limited Broadcasting, and Blocking-ERS which demonstrate its superiority regardless of network size, traffic load, or speed. In TLRDA-C, application data are transmitted earlier due to lower route discovery time and the earlier reception of route replies since no extra delay imposed to the route request dissemination within the source node's neighbourhood region. Furthermore, TLRDA-C reduces the route request latency and routing overhead over TLRDA-D while achieving almost the same end-to-end delay and packet loss.

Chapter 6: Traffic Locality Expanding Ring Search

6.1 Introduction

In on-demand routing protocols, the broadcast of the route request used in route discovery process dominates most of the routing overhead [25, 56, 84, 93] so there is an urgent need to improve this process [131]. The route discovery protocols can be improved to minimise such overhead by stopping the unnecessary propagation of route request packets after the required route has been discovered.

The new approach to traffic locality TLRDA, introduced in Chapter 3, is used in this chapter to develop a new route discovery algorithm called Traffic Locality Expanding Ring Search (TL-ERS) algorithm as an improvement to Expanding Ring Search (ERS) [115]. TL-ERS reduces the route request overhead during the route discovery process by exploiting traffic locality. Since the neighbourhood region includes most of the likely destinations for the source node on hand, broadcasting the route requests first within this region has a very high chance of success. If not, the ring search will be doubled and a second attempt will take place within such ring. If this is unsuccessful, a network wide broadcast is performed. TL-ERS is adaptive and continuously updates the boundaries of the first and second ring to provide better performance.

6.1.1 Expanding Ring Search (ERS)

Network-wide flooding is a very expensive process, thus should be avoided in a resource-limited environment such as MANETs. One way to search for a route without covering the whole network is to use Expanding Ring Search (ERS). It works by searching successively larger areas centred around the source node, until the required route is located. The basic idea behind ERS is to stop the search at the ring where a valid route to the destination is found and avoid flooding the entire network in search of such a route but with a probability of searching the same area more than once. Therefore, the source node starts the search by broadcasting a route request with TTL=

TTL_START to flood the first ring. Each time the source node times out without receiving a reply, it re-initiates the route request with TTL incremented by TTL_INCREMENT. This process continues until a TTL_THRESHOLD is reached. If no route has been located by this time, flooding is used with TTL = network diameter (*D)* and the full network coverage is repeated to a maximum number of retries i.e. in AODV [93] the maximum number of tries is two. All nodes in a connected network use the same fixed predefined values for TTL_START, TTL_INCREMENT, and TTL_THRESHOLD. For instance, ERS is used to improve AODV algorithm described in [80, 93, 94] employing TTL parameters as in Table 6-1. The relation between the rings and TTL is shown in Figure 6-1.

Table 6-1: ERS parameters.

TTL parameter	Value
TTL_START	1
TTL_INCREMENT	2
TTL_THRESHOLD	7

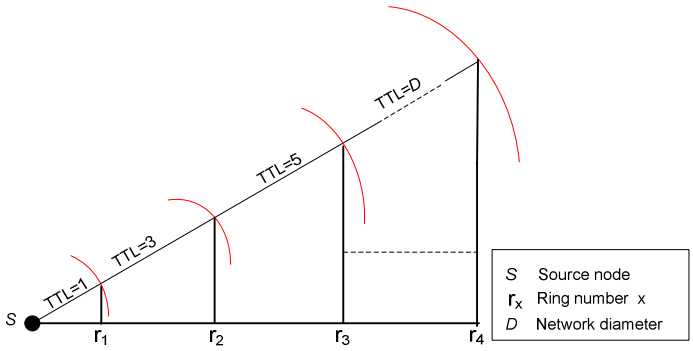

Figure 6-1: Successive rings in ERS

Route Discovery Path (RDP) is the number of hops from the first initiation of a route request until the source node receives the first route reply. Equation (6.1) shows RDP for ERS:

$$RDP_{ERS} = \begin{cases} h_s(f) + \sum_{i=1}^{h_s(f)} (2i-1) & h_s(f) \text{ is odd number} \\ h_s(f) + \sum_{i=1}^{h_s(f)-1} (2i-1) + 1 & h_s(f) \text{ is even number} \end{cases} \quad (6.1)$$

Route Request Path (RRP) is the number of hops that the route request traverses from the first initiation of the route request until it is discarded. Equation (6.2) shows *RRP* for ERS Where

$h_s(f) \leq T$ means success in ring r, $h_s(f) > T$ means last attempt using the whole network coverage, T = TTL-THRESHOLD and r is the ring that contains $h_s(f)$.

$$RRP_{ERS} = \begin{cases} \sum_{i=1}^{r} (2i-1) & h_s(f) \leq T \\ \sum_{i=1}^{r} (2i-1) + D & h_s(f) > T \end{cases} \quad (6.2)$$

Figure 6-2 shows the steps that will be performed by the source node when a route to unknown destination is needed in ERS. If the route request is initiated for the first time, the first step is to assign TTL field to TTL_START (line 8). If the route request is reinitiated then TTL field is incremented by TTL_INCREMENT (line 10) until TTL reaches TTL_THRESHOLD where a complete flooding is done by assigning TTL to the network diameter D (line 5). So the route request is broadcast with the right TTL value (line 13). Another retry is performed using simple flooding if the required route is not found yet.

Algorithm preformed by the source node for initiating or reinitiating route request in ERS.

```
1:  If TTL = D then
2:      destination not found.
3:  else
4:      If TTL > TTL_THRESHOLD then
5:          TTL=D
6:      Else
7:          If first ring then
8:              TTL= TTL_START
9:          Else
10:             TTL= TTL + TTL_INCREMENT
11:         End if
12:     End if
13:     broadcast the route request
14: End if
```

Figure 6-2: TTL initialisation steps for initiating or reinitiating a route request in ERS.

6.2 The Traffic Locality Expanding Ring Search (TL-ERS) algorithm

To obtain a search result that is as close as possible to the optimal search result whilst keeping the cost low, the search strategy has to be set to suit the application scenario and system configuration. ERS is not necessarily better than simple flooding if the ERS parameters are not selected properly [23, 24, 64, 125]. Selecting the initial TTL value for the first search ring is an important step towards a more effective search [125].

Traffic Locality-Expanding Ring Search (TL-ERS) is an improvement of ERS based on TLRDA to utilise the traffic locality concept, introduced earlier in Chapter 3. The main difference between ERS and TL-ERS is in the TTL parameters. ERS uses a fixed radius for all nodes in the network depending on the search ring. However, TL-ERS is adaptive since it uses the value of *LP* as the radius of the first ring where *LP* differs from source node to another and is always updated to reflect the current environment. Here, a detailed performance evaluation of TL-ERS is provided using mathematical and simulation modelling to demonstrate its advantages over the existing Expanding Ring Search (ERS). TL-ERS uses the parameters stated in Table 6-2. Moreover, limiting the number of rings in the worst-case to two or three rings achieves lower cost broadcasting as discussed in [24] thus TL-ERS limits the maximum number of rings to two when $2LP \geq D$ or to three when $2LP < D$ as shown in Figure 6-3 and Figure 6-4 while the maximum number of rings in ERS varies depending on the values used for ERS parameters i.e. sequential ERS, starting from 1, has in the worst-case the highest number of rings because TTL_INCREMENT is 1. The pseudo code for initiating or reinitiating a route request with the correct TTL for TL-ERS is described in Figure 6-4.

Table 6-2: TL-ERS parameters.

TTL parameter	Value
TTL_START	LP
TTL_INCREMENT	LP
TTL_THRESHOLD	2LP

TL-ERS works by initialising the TTL field with the value of *LP* for the first search ring to improve the route discovery path compared to ERS. If the source node times out without receiving a route reply, it reinitiates the route request with TTL equal to twice *LP*. Then if it times out again,

it floods the whole network by assigning TTL to be the network diameter D.

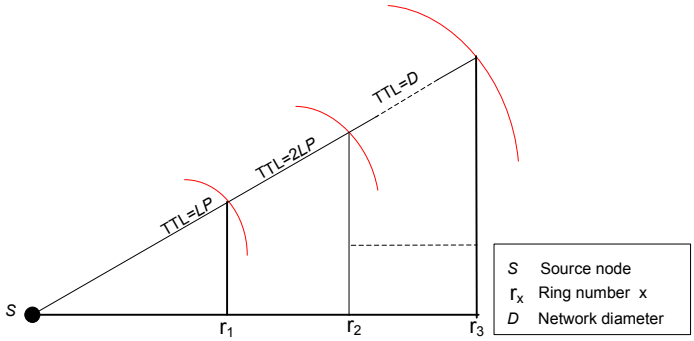

Figure 6-3: Successive rings in TL-ERS.

Figure 6-4 shows the steps that will be performed by the source node when a route to unknown destination is needed in TL-ERS. If the route request is initiated for the first time, the first step is to assign TTL field to LP (line 2). If the route request is reinitiated then TTL field is assigned the value of $2LP$ (line 6). If the source node times out without receiving a route reply after searching the second ring and $2LP < D$, simple flooding is used by setting TTL to the network diameter D (line 10). When the network is completely covered without finding the required route, the algorithm assumes that destination is not found (line 13) where the simple flooding may be retried again several times according to the on-demand routing algorithm used.

ERS and TL-ERS might reduce network overhead but they may increase the route discovery path more than the simple flooding [51, 64] regardless of the parameter values used unless they succeed in the first ring. When route request discovery path increases, the end-to-end delay might increase as well depending on the route request latency per hop.

Algorithm preformed by the source node for initiating or reinitiating route request in TL-ERS.

```
If first ring then
    TTL=LP
    broadcast the route request
Else
    If second ring then
        TTL= 2LP
        broadcast the route request
    Else
        If 2LP < D then
            TTL= D
            broadcast the route request
        Else
            destination not found.
        End if
    End if
End if
```

Figure 6-4: TTL initialisation steps for initiating or reinitiating a route request in TL-ERS.

6.3 Mathematical formulation

Following the mathematical modelling from Chapter 4 and since there is no delay added to route requests propagation in TL-ERS, the total service time of a route request travelling within its neighbourhood or beyond-neighbourhood region is T_{Class1}. Moreover, the equations for the end-to-end delay and route request lifetime for TL-ERS algorithm are derived as follows:

A. Calculating the end-to-end delay

The end-to-end delay can be calculated by adding the route discovery time RDT to the time that the data packet needs to traverse from source node to destination; assuming that data packets have total service time equal to T_{Class1}.

Route discovery path (RDP) as number of hops is shown in Equation (6.3) for TL-ERS:

$$RDP_{TL-ERS} = \begin{cases} 2h_s(f) & h_s(f) \leq LP \\ 2h_s(f) + LP & LP < h_s(f) \leq 2LP \\ 2h_s(f) + 2LP & 2LP < h_s(f) \leq D \end{cases} \quad (6.3)$$

To illustrate the difference in RDP between TL-ERS, ERS, and AODV, let us assume the finder of the route is four hops away from the source node as illustrated in Figure 6-5. AODV discovers the required route after four hops. However, TL-ERS discovers it after six hops assuming LP = 2 while ERS traverses eight hops to discover the same route. So *RDP* is 8, 10, 12 hops for AODV, TL-ERS, and ERS respectively. On the other hand, incurring lower network overhead reduces congestion level and lowers channel contention. This in turn reduces the route request latency per hop having a positive impact on the route discovery time and improving the end-to-end delay.

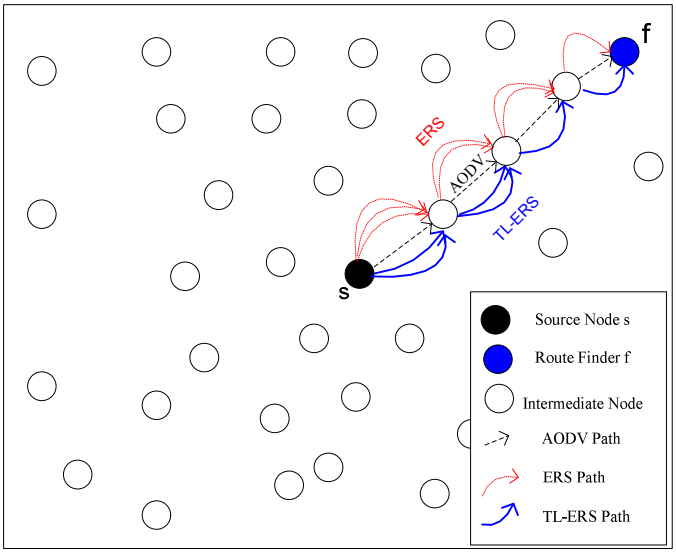

Figure 6-5: The route discovery path for TL-ERS, ERS and AODV when the finder of the required route is four hops away from the source node.

Route discovery time (RDT) can be calculated as:

$$RDT = RDP_{TL-ERS} T_{Class1} \tag{6.4}$$

So the end-to-end delay can be calculated as follows:

$$\text{End} - \text{to} - \text{end delay} = RDT + h_s(d) T_{Class1} \tag{6.5}$$

B. Calculating the route request lifetime (RRL)

The route request lifetime *(RRL) is* the time from sending a particular route request by the source node for the first time until such route request is discarded due to the success of the current search. To calculate the *RRL*, we need to calculate route request path (*RRP)* first.

Equation (6.6) shows *RRP* of TL-ERS for $h_s(f) \leq LP$, $LP < h_s(f) \leq 2LP$, or $2LP < h_s(f) \leq D$ corresponding to success in first ring, second ring, or whole network coverage respectively.

$$RRP_{TL-ERS} = \begin{cases} LP & h_s(f) \leq LP \\ LP + 2LP & LP < h_s(f) \leq 2LP \\ LP + 2LP + D & 2LP < h_s(f) \leq D \end{cases} \quad (6.6)$$

$$RRL = RRP_{TL-ERS} T_{Class1} \quad (6.7)$$

6.3.1 Comparison between TL-ERS and ERS

All packets in both algorithms TL-ERS and ERS belong to Class 1 assuming that $T_{Class1} = 1$. The hop counts are 2, 3… 10 for different sources and route finders under the same environment. Such hop counts correspond to the network sizes 20, 30… 100 nodes divided by 10.

Figure 6-6 shows that end-to-end delay increases with the increment of network size for both TL-ERS and ERS algorithms due to the increment in hop count. However, TL-ERS discovers new routes quicker than ERS since it requires less number of rings to find the same routes. When the first ring is large enough to contain the finder of the route, the discovery time is improved leading to better end-to-end delay.

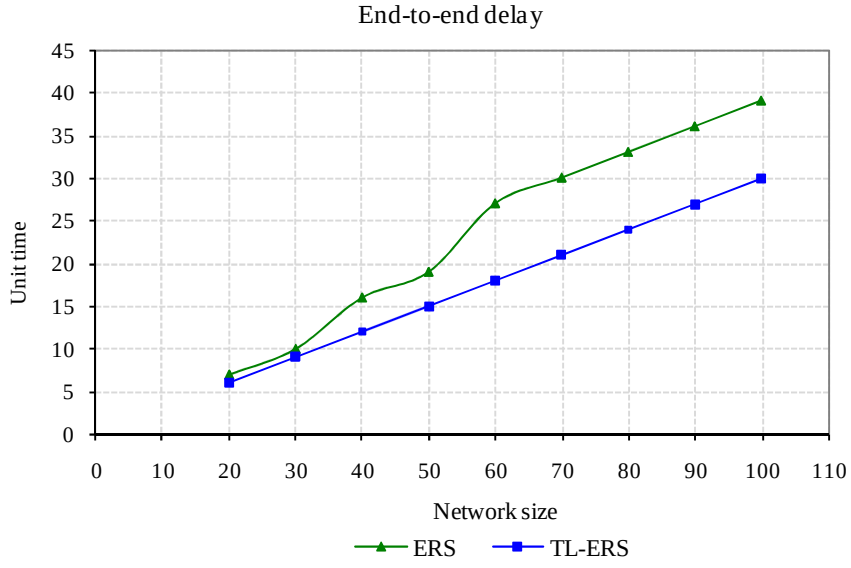

Figure 6-6: End-to-end delay versus network size when $h_s(f) \leq LP_r$.

To consider the success in the second ring for TL-ERS, the values of *LP* are calculated as $h_s(f) > LP$. Figure 6-7 shows that when the hop count to route finder is larger than 3, TL-ERS discovers new routes quicker than ERS otherwise their end-to-end delays are almost the same.

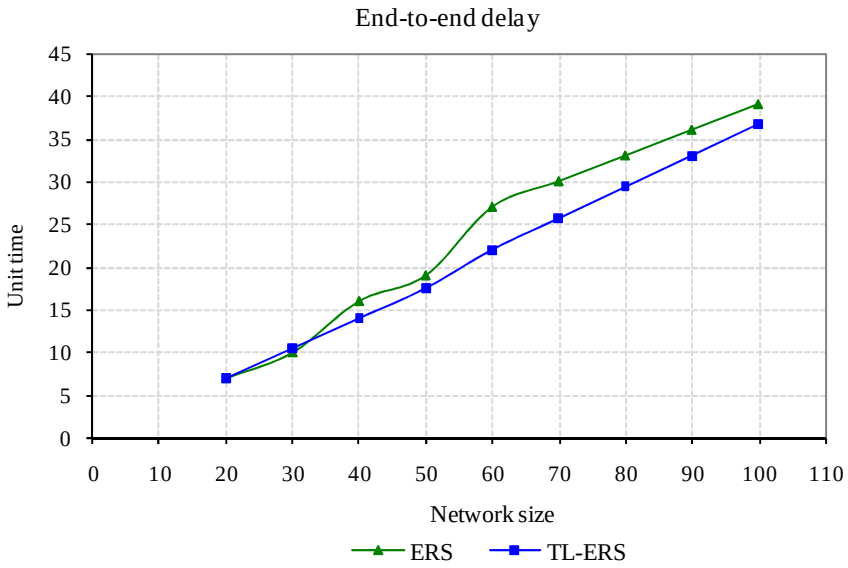

Figure 6-7: End-to-end delay versus network size when $h_s(f) > LP_r$.

6.4 Simulation

Simulation has been conducted to evaluate TL-ERS against simple flooding, referring to it as AODV, and the Expanding Ring Search [94], referring to it as ERS. TL-ERS was implemented as a modification to AODV implementation on ns2 version 2.29 [41]. Extensive experiments were conducted to evaluate the performance of TL-ERS and compared it with both AODV and ERS. The comparison metrics include the route request latency and end-to-end delay to study network latency. Also they include routing overhead to study network overhead and packet loss to study congestion level. Moreover, the simulation analysis considers all the three analyses cases, network size, traffic load, and maximum speed as stated earlier in Chapter 2 as in Table 2-3.

6.4.1 Effect of network size

Figure 6-8 to Figure 6-12 display the results of running our algorithm, TL-ERS, against AODV and ERS using networks with different number of nodes increased as multiple of 10 starting from 20 to 100 with a minimum speed of 1m/s and a maximum speed of 15m/s. The number of communication sessions is fixed to ten.

Latency:

ERS and TL-ERS might increase the discovery time more than AODV if they do not succeed in the first ring. The end-to-end delay is a very important measurement because it includes the discovery time where application data are queued in the source node for that time. In ERS and TL-ERS, the hop counts for the discovery path increases with each repeated ring. In contrast, ERS and TL-ERS free the network from fulfilled route request especially if succeeded in early rings reducing congestion and channel contention which improves the discovery time but delaying the discovery when repeating the search increases the discovery time reusing the improvement that is due to reduction in channel contention.

Figure 6-8 shows that end-to-end delay increases with the increment of network size regardless of the algorithm used. Moreover, TL-ERS outperform both ERS and AODV in terms of end-to-end delay. Specifically, TL-ERS improves end-to-end delay by 26% to 44% over AODV and up to 38% over ERS.

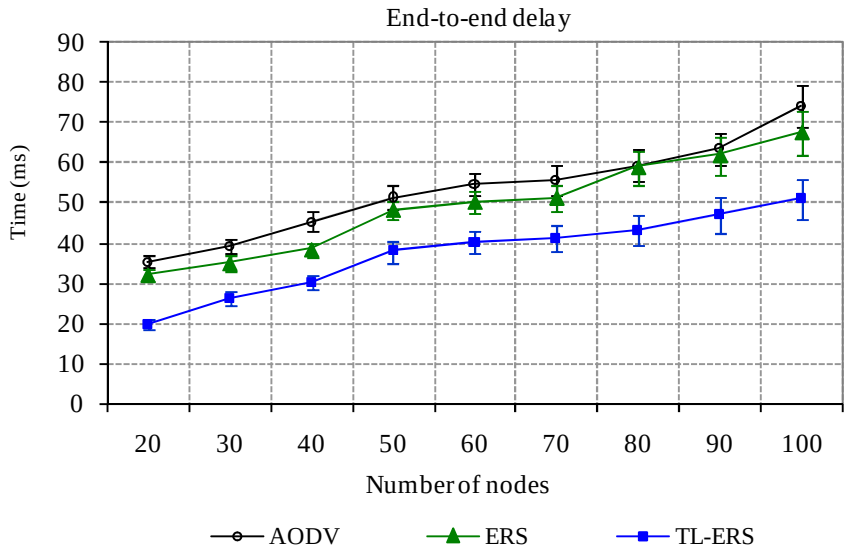

Figure 6-8: End-to-end delay verses network density with 10 communication sessions and 15m/s as maximum speed.

The average of route request latency per hop is almost the same for both TL-ERS and ERS as shown in Figure 6-9. Moreover, AODV covers the whole network due to simple flooding so route requests reside in the network for longer time which makes them prone to link breakage and higher channel contention.

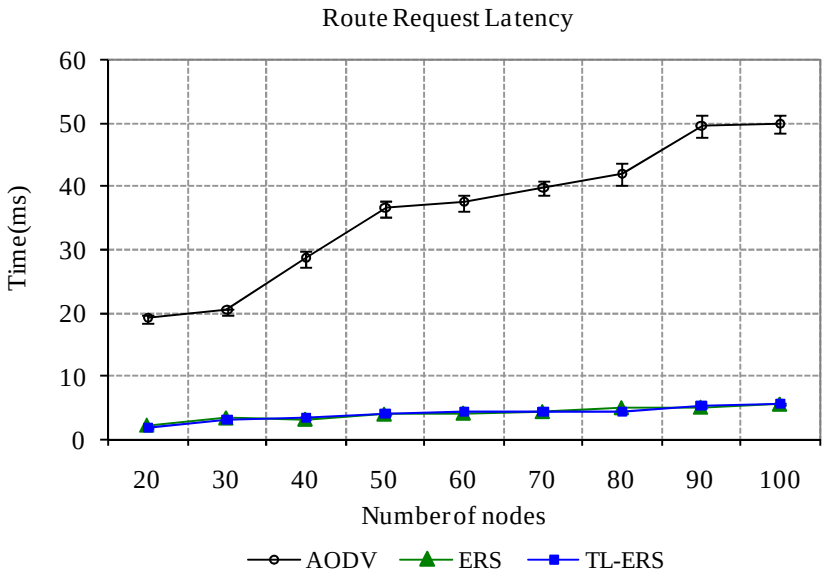

Figure 6-9: Route request latency verses network size with 10 communication sessions and 15m/s as maximum speed.

When route requests propagate in the network, they share resources with other packets which add to the network congestion and increases channel contention delaying route requests more. TL-ERS improves the average of route request latency by 90% over AODV and 17% over ERS in small size networks while its improvement in moderate size networks is 89% over AODV and no improvement over ERS.

Overhead:

Figure 6-10 demonstrates the improvements of TL-ERS over ERS and AODV by minimising the route request overhead. The improvement in routing overhead for both ERS and TL-ERS over AODV is due to the limiting propagation of route requests to the ring that contains the finder of the required route but with the risk of visiting this area more than once in the event of unsuccessful search.

For this reason, route request overhead might be reduced but causing end-to-end delay to increase. TL-ERS improves routing overhead by 89% to 98% over AODV and by 18% to 42% over ERS. To clearly show the difference between ERS and TL-ERS, we magnified the lower part of Figure 6-10 up to 350 route requests and demonstrated it in Figure 6-11.

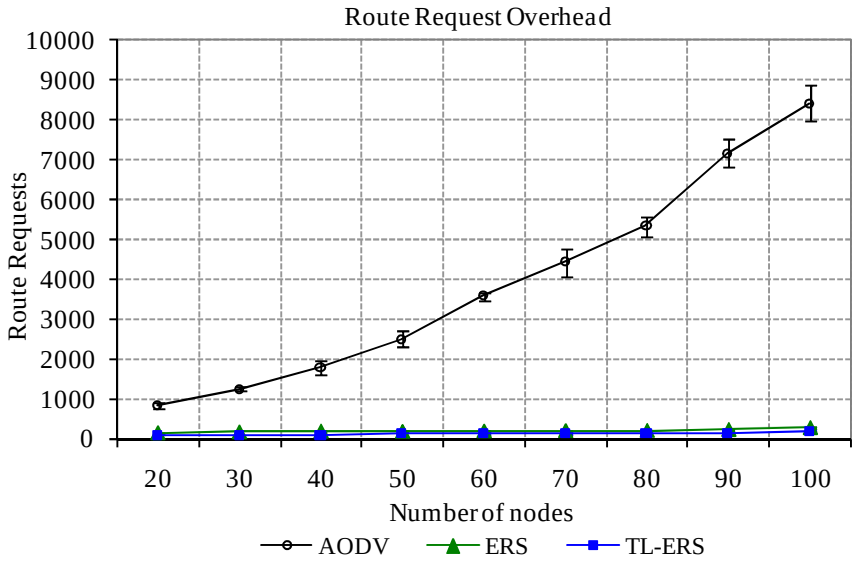

Figure 6-10: Routing overhead versus network size with 10 communication sessions and 15m/s as maximum speed; comparing TL-ERS with AODV and ERS.

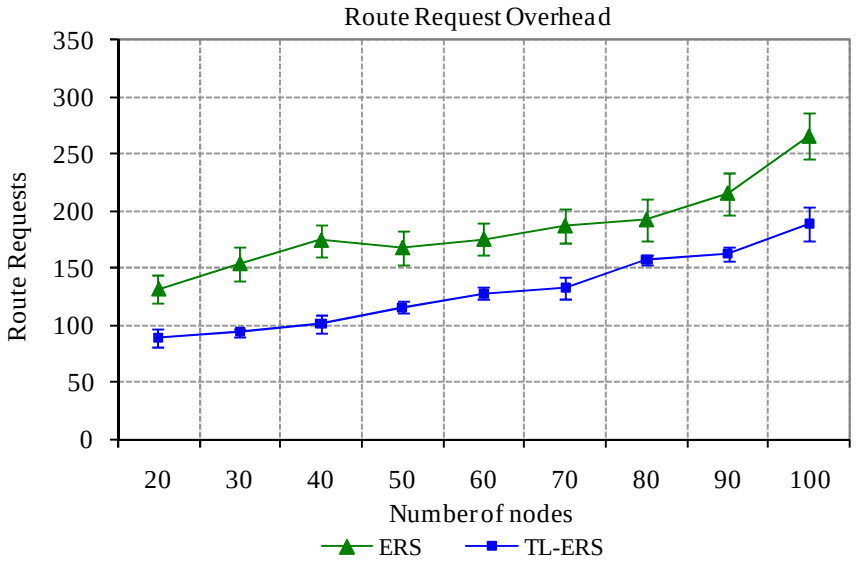

Figure 6-11: Routing overhead versus network size with 10 communication sessions and 15m/s as maximum speed; comparing TL-ERS with ERS.

TL-ERS incurs lower routing overhead than ERS which means TL-ERS repeats the search less number of times. So in TL-ERS, network performance improves due to the reduction in the number of received route requests as presented in Figure 6-10 without increasing end-to-end delay as depicted in Figure 6-8. This has a generally beneficial effect on the network performance due to the fact that the data can typically travel earlier and with less congestion.

Congestion:

ERS and TL-ERS limit the broadcast of route requests as opposed to simple flooding in AODV. To this end, both ERS and TL-ERS reduce congestion and channel contention which improves packet loss in such networks. Figure 6-12 shows that TL-ERS improves packet loss by 26% to 85% over AODV. Most of the time, TL-ERS improves packet loss more than ERS while in few situations ERS loses a little fewer packets.

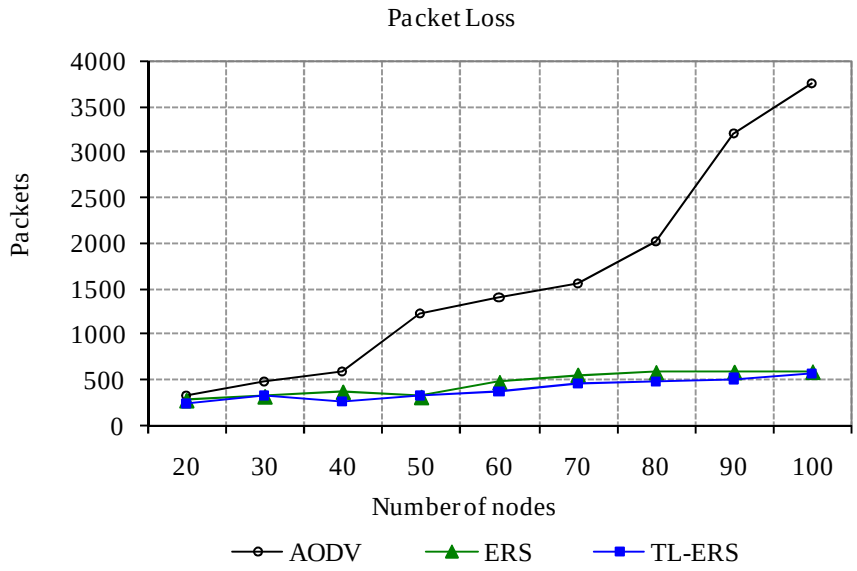

Figure 6-12: Packet loss versus network size with 10 communication sessions and 15m/s as maximum speed.

6.4.2 Effect of traffic load

Figure 6-13 to Figure 6-16 display the results of running our algorithm, TL-ERS, against ERS and AODV using networks of size 70 nodes with a random speed ranging between 1 and 15m/s. The traffic load ranges from 5 to 35 communication sessions incremented by five.

Latency:

Figure 6-13 shows that end-to-end delay increases with the increment of traffic load in all three algorithms. This figure reveals the fact that TL-ERS reduces end-to-end delay more than both ERS and AODV. ERS and TL-ERS reduce congestion and channel contention compared to AODV because they both control the propagation of route requests whilst AODV covers the whole network. On the other hand, TL-ERS reduces end-to-end delay more than ERS due to the customised values of parameters that is specific to each source node i.e. TTL_START. Choosing

the value for TTL_START carefully increases the chance of success search in the first ring. TL-ERS improves end-to-end delay by 26% to 38% over AODV and by 19% to 32% over ERS.

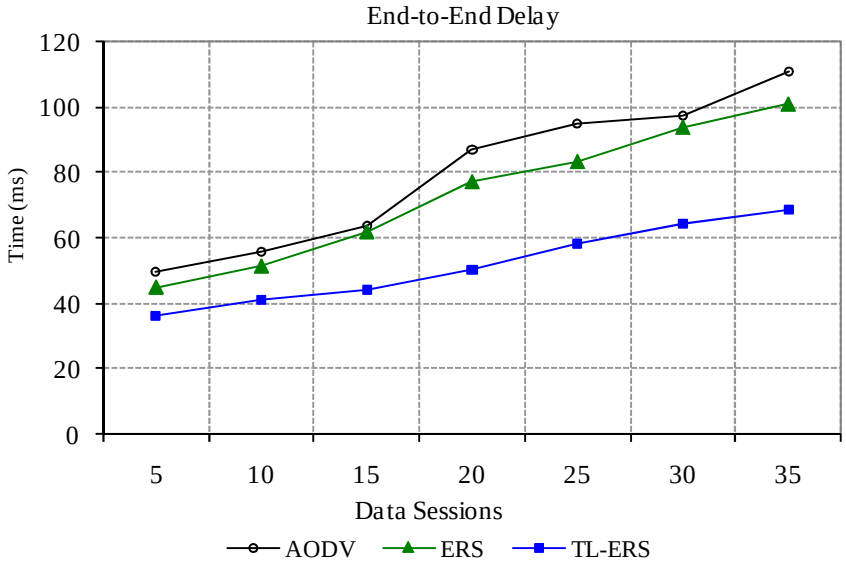

Figure 6-13: End-to-end delay versus traffic load in networks of 70 nodes and 15m/s as maximum speed.

Route request latency increases with more traffic load due to the increment in the number of route discoveries needed as shown in Figure 6-14 with all the algorithms used. This figure also demonstrates that TL-ERS improves route request latency over ERS and AODV especially in heavy traffic. ERS and TL-ERS improve route request latency over AODV due to controlling the route requests dissemination so the route request does not cover the whole network unless all previous attempts failed. TL-ERS improves route request latency by 83% to 92% over AODV and by up to 53% over ERS.

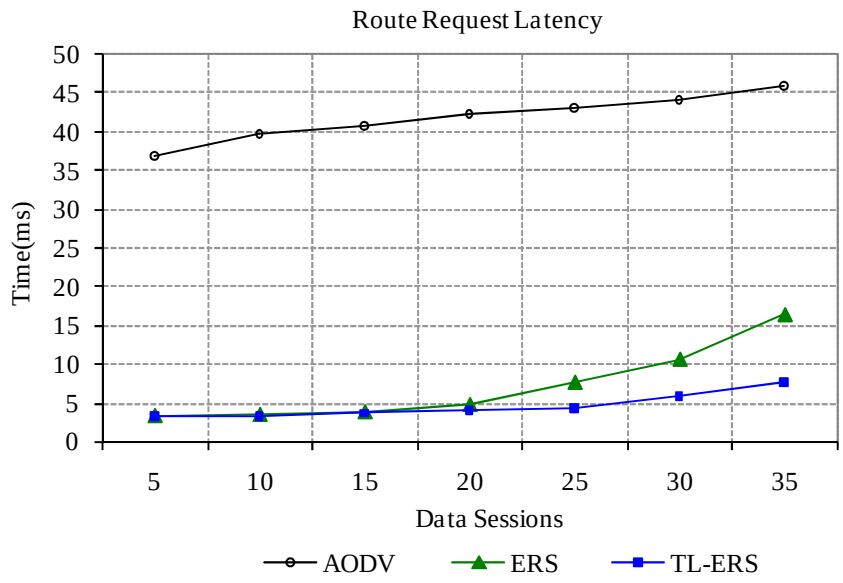

Figure 6-14: Route request latency versus traffic load in networks of 70 nodes and 15m/s as maximum speed.

Overhead:

Route request overhead was increased with heavy traffic due to the increment in number of different route requests for all three algorithms as shown in Figure 6-15.

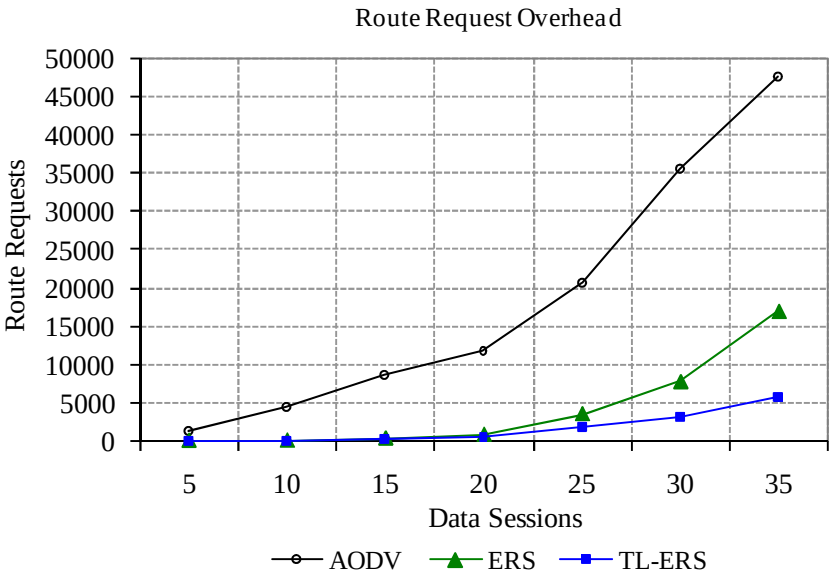

Figure 6-15: Routing overhead versus traffic load in networks of 70 nodes and 15m/s as maximum speed.

ERS and TL-ERS incur lower overhead than AODV because the dissemination of route requests was controlled. However, the number of route request received in TL-ERS is less than those of ERS, knowing that they both need to cover the same area to discover the same route, which means that TL-ERS goes through less number of rings than ERS. Due to that, TL-ERS improves route

request overhead by 89% to 97% over AODV and by up to 65% over ERS.

Congestion:

ERS and TL-ERS incur low overhead because they work in less congested network which reduces channel contention. For this reason, packet loss is reduced in both algorithms as shown in Figure 6-16. However with heavy traffic scenarios, TL-ERS loses fewer packets than ERS. TL-ERS reduces packet loss by 54% to 82% over AODV and up to 51% over ERS depending on the traffic load.

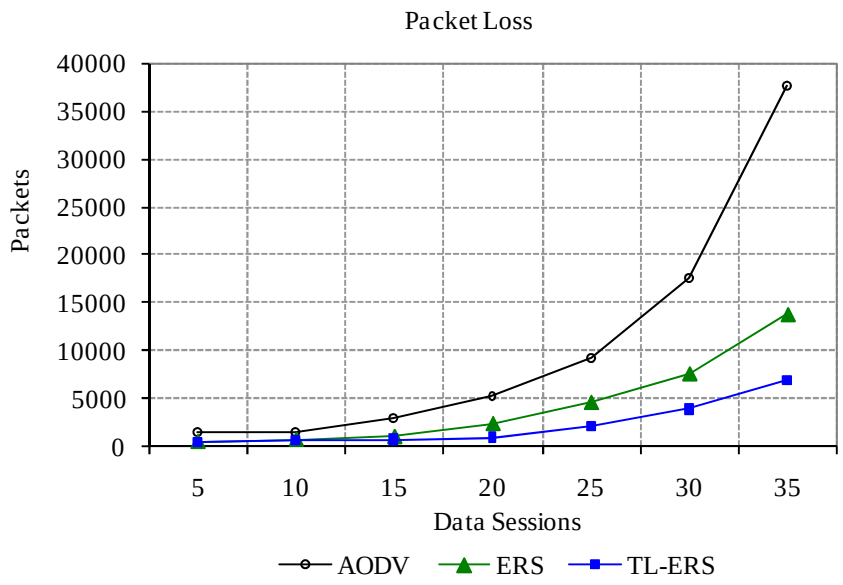

Figure 6-16: Packet loss versus traffic load in networks of 70 nodes and 15m/s as maximum speed.

6.4.3 Effect of mobility

Figure 6-17 to Figure 6-21 were extracted from the simulation runs for TL-ERS, ERS and AODV while increasing the maximum speed starting from 2 to 15m/s and injecting 10 communication sessions in networks of 70 nodes.

Latency:

The end-to-end delay increases with the fast speed networks in all the three algorithms because the speed effects the routes and may result in broken links as shown in Figure 6-17. TL-ERS improves end-to-end delay more than AODV by 22% to 27% due to the reduction in network congestion. Moreover, TL-ERS improves end-to-end delay more than ERS by 14% to 19% because the first ring in TL-ERS might cover more nodes.

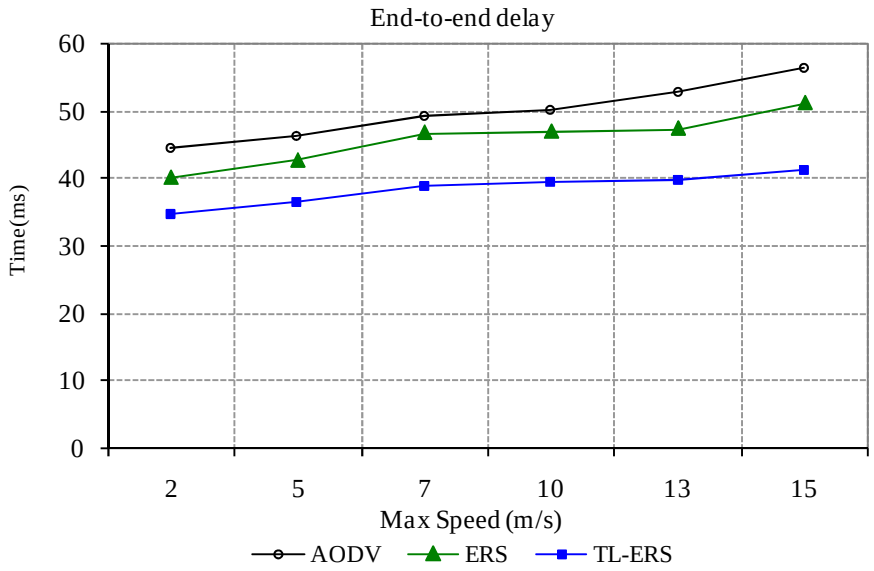

Figure 6-17: End-to-end delay versus maximum speed in networks of 70 nodes, and 10 communication sessions.

Figure 6-18 shows that TL-ERS gives slight improvement in route request latency compared to ERS regardless of speed because both algorithms broadcast route request without delaying its propagation. On the other hand, ERS and TL-ERS improve the route request latency over AODV due to the controlled propagation of route requests. TL-ERS improves route request latency by 90% to 91% over AODV.

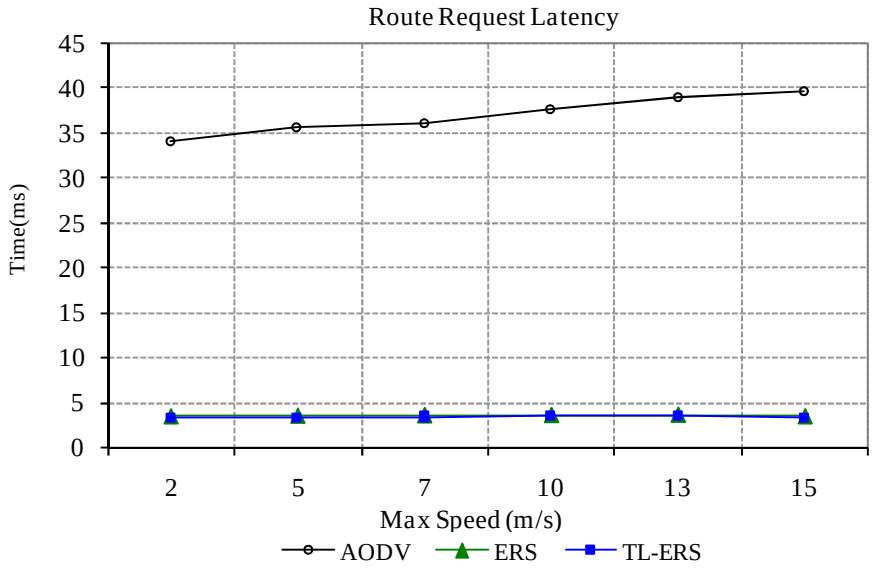

Figure 6-18: Route request latency versus maximum speed in networks of 70 nodes, and 10 communication sessions.

Overhead:

Figure 6-19 demonstrates the superiority of both TL-ERS and ERS over AODV by minimising the route request overhead. The improvement in routing overhead in both ERS and TL-ERS over AODV is due to the controlled propagation of route requests opposed to full network coverage. TL-ERS improves routing overhead by 96% to 97% over AODV.

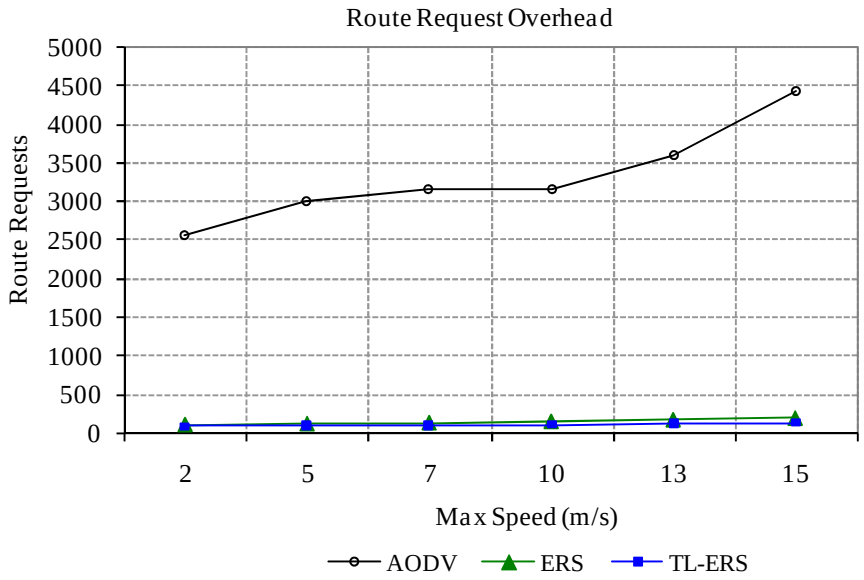

Figure 6-19: Routing overhead versus maximum speed in networks of 70 nodes and 10 communication sessions; comparing TL-ERS with AODV and ERS.

To clearly show the difference in routing overhead between ERS and TL-ERS, we magnified the lower part of Figure 6-19 by scale of 200 rather than 5000 route requests as demonstrated in Figure 6-20. TL-ERS incurs lower routing overhead than that of ERS which means TL-ERS repeats the search less number of times thus reduces routing overhead by 20% to 29% over ERS and 96% to 97% over AODV as stated earlier.

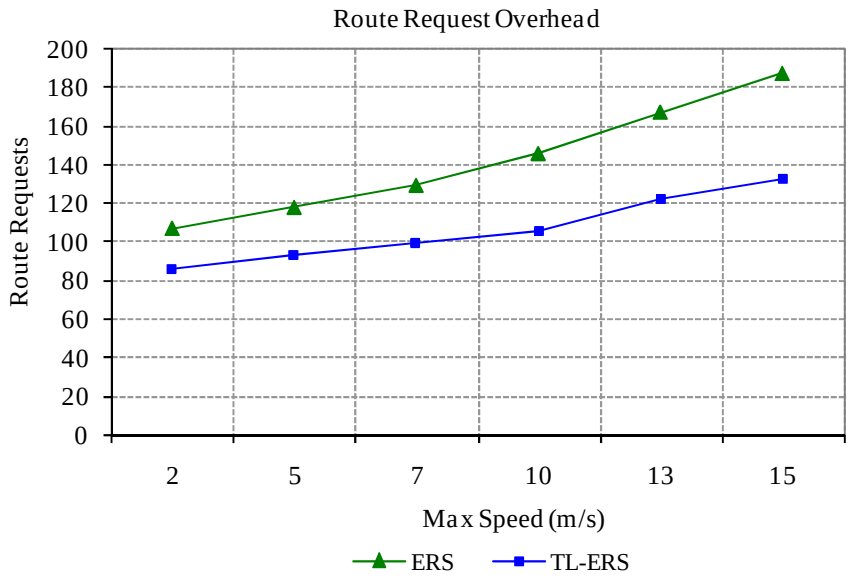

Figure 6-20: Routing overhead versus maximum speed in networks of 70 nodes and 10 communication sessions; comparing TL-ERS with ERS.

Congestion:

As ERS and TL-ERS improve routing overhead over AODV which improves network congestion and channel contention leading to improvement in packet loss. TL-ERS improves packet loss by 60% to 75% over AODV and up to 17% over ERS.

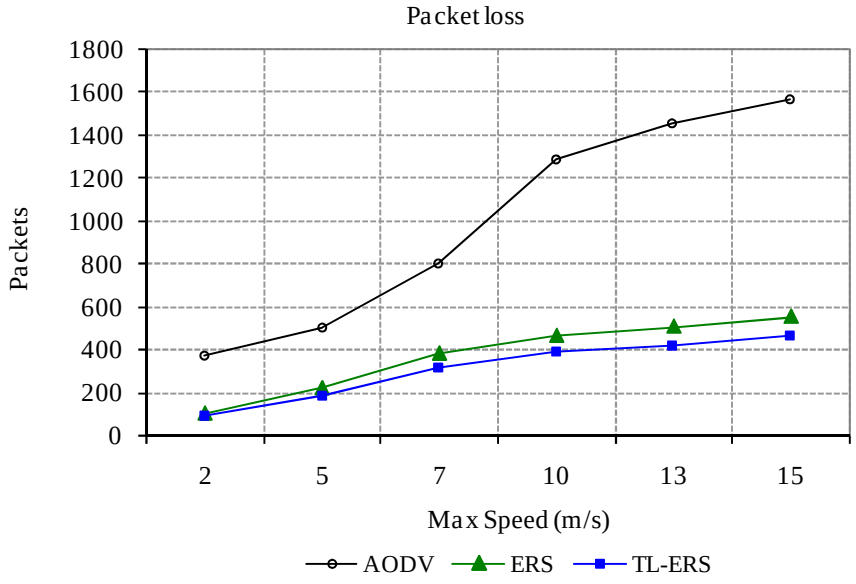

Figure 6-21: Packet loss versus maximum speed in networks of 70 nodes and 10 communication sessions.

So in TL-ERS, network performance improves due to the reduction in the number of received route requests as presented in Figure 6-20 while improving end-to-end delay as depicted in Figure

6-17 which has a generally beneficial effect on the network performance due to the fact that the data can typically travel earlier and with less congestion.

6.5 Summary of simulation results

Simulation experiments and performance analyses were conducted for TL-ERS from network size, traffic load, and mobility prospective. The same trend was observed when comparing the performance of TL-ERS with that of both ERS and AODV as depicted in Figure 6-8 to Figure 6-21. The percentages of TL-ERS improvement over both AODV and ERS according to network size, traffic load, and mobility are stated in Table 6-3. Moreover, such improvement is presented from route request latency, end-to-end delay, route request overhead, and packet loss prospective. To simplify the table, small and moderate size networks for the network size case, light and heavy traffic for traffic load case, and slow and fast networks for the mobility case are shown.

Table 6-3: Summary of the improvements for TL-ERS over both AODV and ERS.

Cases	Algorithm	Route Request Latency		End-to-end delay		Routing Overhead		Packet Loss	
Network size		Small	Moderate	Small	Moderate	Small	Moderate	Small	Moderate
	AODV	90%	89%	44%	38%	89%	98%	26%	85%
	ERS	17%	0%	31%	24%	33%	29%	10%	3%
Traffic load		Light	Heavy	Light	Heavy	Light	Heavy	Light	Heavy
	AODV	91%	83%	26%	38%	89%	88%	66%	82%
	ERS	2%	53%	19%	32%	8%	65%	1%	51%
Mobility		Slow	Fast	Slow	Fast	Slow	Fast	Slow	Fast
	AODV	90%	91%	22%	27%	97%	97%	75%	70%
	ERS	1%	1%	14%	19%	20%	29%	12%	16%

ERS and TL-ERS improve network performance over AODV in terms of: latency, overhead, and congestion. However, TL-ERS improves network performance compared to ERS due to the reduction in the route request overhead as well as end-to-end delay. The attractiveness of this improvement stems from the fact that the data can travel earlier with less congestion.

6.6 Comparison of TL-ERS with TLRDA-D and TLRDA-C

Nodes in TLRDA-D, introduced in Chapter 4, broadcast route request that is travelling within its source node's neighbourhood region according to the routing algorithm used while they broadcast the route request that is travelling within its source node's beyond-neighbourhood region with a delay equal to $2LP_r * NTT$.

TLRDA-C, introduced in Chapter 5, utilises the chase packet concept to improve TLRDA-D. When a source node receives a route reply as an answer to its query in TLRDA-C, it transmits a chase packet to catch and terminate the fulfilled route request. The chase packet travels faster than the route request in the beyond-neighbourhood region in order to increase the success rate of the catching process.

The data extracted from the simulation runs show that both TLRDA-D and TLRDA-C succeed in improving the discovery time leading to improvement in the end-to-end delay. Moreover, TL-ERS improves average route request latency and overhead. The effect of network size on TLRDA-D, TLRDA-C, and TL-ERS are compared below using the results from Sections 4.5.1, 5.4.1, and 6.4.1 respectively.

Latency:

TLRDA-D and TLRDA-C improve network congestion and reduce channel contention; so both have almost the same end-to-end delay since the difference is negligible. However, TL-ERS has higher end-to-end delay as shown in Figure 6-22 due to the increase of discovery time if the search is not successful in the first ring. TLRDA-D and TLRDA-C improve end-to-end delay by up to 109% over TL-ERS.

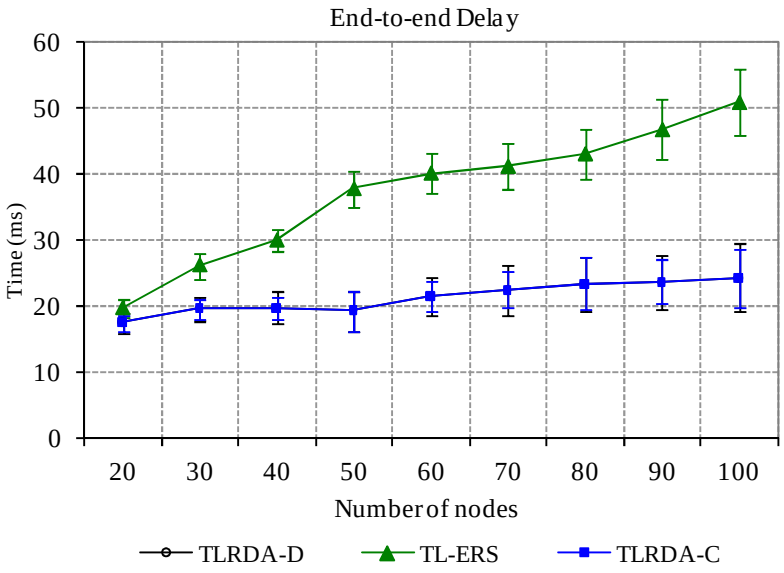

Figure 6-22: End-to-end delay versus network size with 10 communication sessions and 15m/s as maximum speed.

Figure 6-23 shows the success of TLRDA-C and TL-ERS in minimising the average of route request latency. The average route request latency of TLRDA-C is reduced because the catching process discards unwanted route requests in the beyond-neighbourhood region.

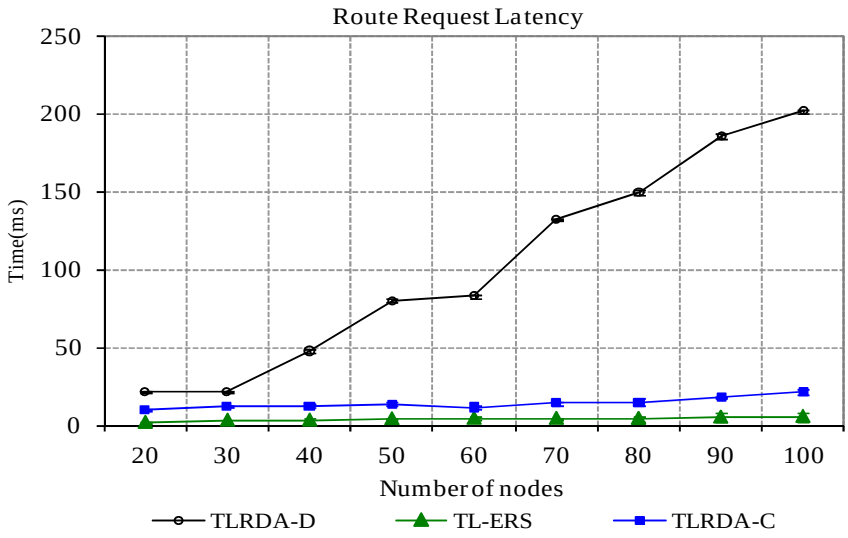

Figure 6-23: Average route request latency versus network size with 10 communication sessions and 15m/s as maximum speed.

TL-ERS improves the route request latency because it adds no delay to the route request propagation and avoids further propagation of the route request. TL-ERS improves the average of route request latency by 85% to 97% over TLRDA-D and by 63% to 74% over TLRDA-C.

Overhead:

TL-ERS incurs a lower routing overhead than both TLRDA-D and TLRDA-C as shown in Figure 6-24 because TL-ERS adds no new control packets and limits the broadcast to small area. The success of the catching process in TLRDA-C frees fulfilled route requests thus reduces overhead compared to TLRDA-D. Moreover, TL-ERS improves the average routing overhead by 89% to 98% over TLRDA-D and by 87% to 95% over TLRDA-C.

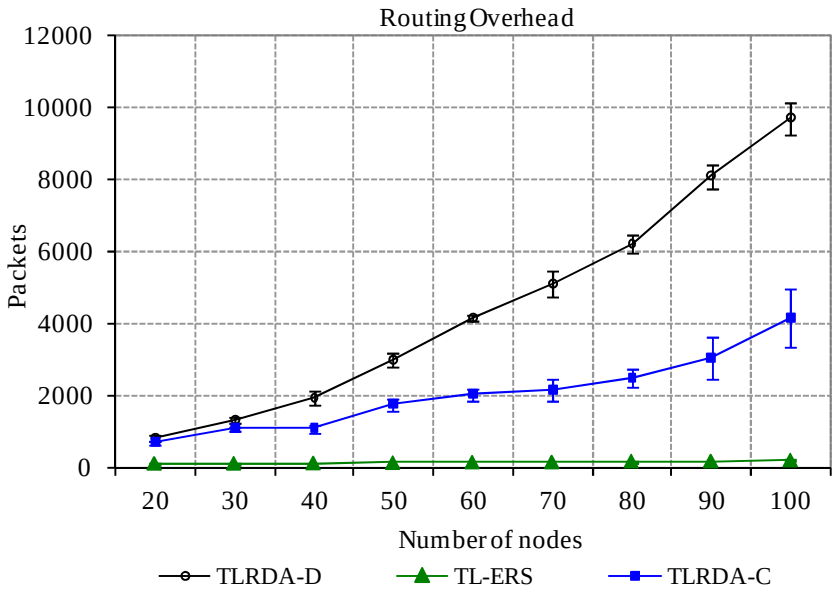

Figure 6-24: Routing overhead versus network size with 10 communication sessions and 15m/s as maximum speed.

Congestion:

TL-ERS losses fewer packets than both TLRDA-D and TLRDA-C especially in moderate size network as shown in Figure 6-25 because TL-ERS limits the broadcast to small area avoiding congesting the network with unnecessary flooding. TL-ERS increases the packet loss more than TLRDA-D and TLRDA-C in small size networks. However, it reduces the packet loss in moderate size networks by 60% over TLRDA-D and by 57% over TLRDA-C.

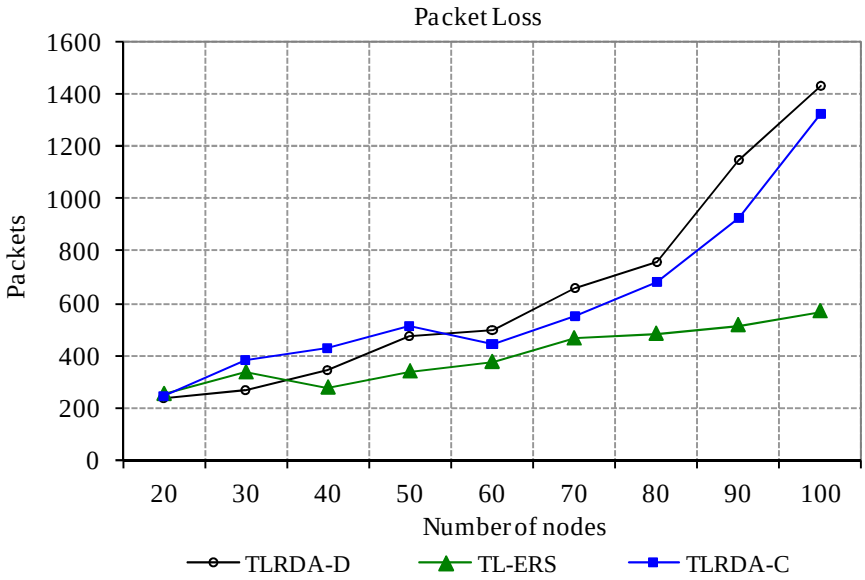

Figure 6-25: Packet loss versus network size with 10 communication sessions and 15m/s as maximum speed.

The effects of the traffic load and mobility for TLRDA-D, TLRDA-C, and TL-ERS show the same behaviour as the effect of network size. The results of the comparison between the three algorithms are summarised in Table 6-4 in term of routing overhead and in Table 6-5 in term of end-to-end delay.

Table 6-4: Routing overhead improvement for TL-ERS and TLRDA-C over TLRDA-D.

Cases	Algorithm	Routing Overhead	
Network size		Small	Moderate
	TL-ERS	89%	98%
	TLRDA-C	17%	57%
Traffic load		Light	Heavy
	TL-ERS	93%	89%
	TLRDA-C	67%	83%
Mobility		Slow	Fast
	TL-ERS	97%	97%
	TLRDA-C	66%	58%

Table 6-5: End-to-end delay improvement for TLRDA-D and TLRDA-C over TL-ERS.

Cases	Algorithm	End-to-end delay	
Network size		Small	Moderate
	TLRDA-D	12%	109%
	TLRDA-C	12%	109%
Traffic load		Light	Heavy
	TLRDA-D	41%	40%
	TLRDA-C	41%	44%
Mobility		Slow	Fast
	TLRDA-D	88%	98%
	TLRDA-C	69%	66%

In summary, comparing TLRDA-D, TLRDA-C, and TL-ERS demonstrates the success of TLRDA-C and TLRDA-D in reducing the end-to-end delay which improves the discovery process. However, TL-ERS and TLRDA-C succeed in reducing routing overhead over TLRDA-D.

6.7 Conclusions

The new approach to route discovery TLRDA, introduced earlier in Chapter 3, was used to develop a new route discovery algorithm, referred to as Traffic Locality Expanding Ring Search (TL-ERS). This algorithm improves the route discovery process in applications that exhibit traffic locality for MANETs in terms of latency and overhead compared to AODV and ERS.

TL-ERS works by establishing a neighbourhood that includes the most likely destinations for a particular source node then broadcasts route requests using this neighbourhood as a first locale or ring, in which to search for the target. If route discovery in this ring proves unsuccessful, the algorithm then establishes a second ring, double the size of the first, and if route discovery fails again the algorithm finally resorts to flooding. Moreover, TL-ERS is adaptive and continuously updates the boundary of the source node's neighbourhood to improve performance and sets the maximum number of rings to three to improve the worst-case performance.

A performance evaluation for TL-ERS was conducted to compare it with the Expanding Ring Search (ERS) algorithm and AODV (with simple flooding). TL-ERS and ERS improve network performance over AODV in terms of latency and overhead. However, the evaluation has shown

that TL-ERS exhibits lower route request overhead and reduces end-to-end delay compared to ERS due to minimising the number of rings needed to search. The low end-to-end delay and routing overhead in TL-ERS have a positive impact on network performance since the transmission of data packets starts earlier due to the former and with less congestion due to the latter.

Chapter 7: Conclusions and Future Work

7.1 Introduction

The increased popularity of wireless devices has brought the potential application promises of Mobile Ad hoc Networks (MANETs) closer to reality [4, 84, 93]. As a consequence, MANETs have been the subject of intensive research over the recent few years, [2, 4, 26, 27, 29, 45, 84, 93, 117, 119]. This is because existing protocols and mechanisms for infrastructure networks cannot be used for MANET without appropriate modifications [56, 84, 93, 117] due to their inherently different characteristics such as mobility, limited power, and the wireless nature of the shared medium.

A major challenge in MANETs is the design of an efficient routing protocol that can accommodate their dynamic nature due to the frequent topology changes. To this end, a number of routing algorithms have been proposed [3, 49, 60, 94, 130]. Broadcasting is an essential component of on-demand routing protocols as it is used for broadcasting route requests to discover new routes between a given source-destination pair. Existing on-demand routing protocols depend on the conventional *simple flooding* for broadcasting which may lead to the well-known broadcast storm problem [131]. A number of research studies have addressed broadcasting in MANETs [6, 8, 20, 24, 44, 71, 79] to try to alleviate this problem. To improve the performance of the route discovery process, broadcast of route requests should be controlled by avoiding the full network coverage [20, 23, 38, 44, 66, 89]. Limiting the broadcast improves network performance by reducing communication overhead and congestion levels. The aim of this research is to propose new algorithms to improve route discovery process in on-demand routing protocol.

7.2 Summary of contributions

This research has proposed, developed, and analysed several new algorithms for improving route discovery process for on-demand routing protocols in MANETs. The major contributions are summarised below.

Traffic locality oriented Route Discovery Approach

Traffic locality oriented Route Discovery Approach (TLRDA) has been introduced then used as the base for the development of our new route discovery algorithms for MANETs which run applications that exhibit traffic locality. TLRDA works by establishing a neighbourhood region for each active source node that includes the most likely destinations. Nodes broadcast a route request without adding any extra delay within that route request neighbourhood region to improve the process of route discovery in on-demand routing protocols.

Traffic Locality oriented Route Discovery Algorithm with Delay

Traffic Locality oriented Route Discovery Algorithm with Delay (TLRDA-D) is a new route discovery algorithm with delay that is based on TLRDA. Each node in TLRDA-D broadcasts a route request according to the on-demand routing algorithm used while it is propagating within its source node's neighbourhood region. Beyond that, the route request is broadcast with a delay in its source node's beyond-neighbourhood region until such broadcast fades away as either the TTL field reaches zero or the connected network is fully covered. The reasoning behind adding this delay is to give route requests that are travelling within their own source node's neighbourhood region priority since route requests travelling in their source node's beyond-neighbourhood region have a higher probability of being already fulfilled. This delay improves the congestion level of the whole network and has been studied using mathematical and simulation modelling.

Several simulation experiments have been performed to study TLRDA-D and compare its performance with that of simple flooding used in AODV [94]. The simulation environments consist of different network scenarios with various network size, traffic load, and maximum speed under the RPGM model. Several instances of TLRDA-D were implemented using logarithmic, linear, or polynomial delay. Our results showed that the best performance among all instances of TLRDA-D was achieved when the delay was set to double the depth of the source node's neighbourhood region. This algorithm improves the end-to-end delay because route requests are broadcast without any delay within their own source node's neighbourhood region in a less congested environment as explained above. For instance when varying network size, TLRDA-D improved the end-to-end delay by up to 67% and reduced packet

loss by up to 62% with no more than 15% increment in routing overhead.

Traffic Locality oriented Route Discovery Algorithm with Chase

Traffic Locality oriented Route Discovery Algorithm with Chase (TLRDA-C) is a new route discovery algorithm that introduces the chase packet concept into TLRDA-D to improve routing overhead without negative impact on the end-to-end delay. Upon receiving a route reply, the source node transmits a chase packet to catch and terminate the original route request. The chase packet is intended to terminate the further propagation of the fulfilled route request as close as possible to the boundary of its neighbourhood region. This is possible because the chase packet travels faster than the route request in the beyond-neighbourhood region, the route request having been deliberately subjected to an artificial delay in this region.

Numerous simulation experiments have been carried out to study TLRDA-C and compare its performance with that of simple flooding used in AODV [94]. TLRDA-C has also been compared with two other algorithms that utilise chase packet concept namely Limited Broadcasting (L-B) [44] and Blocking-ERS (B-ERS) [89]. The simulation environments have considered different network scenarios scrutinised according to network size, traffic load, and maximum speed under the RPGM model. Our performance results revealed that TLRDA-C outperforms L-B, B-ERS, and AODV in terms of the success rate of the catching process, end-to-end delay, route request latency, routing overhead, and packet loss. For instance, when varying network size the end-to-end delay improvement was up to 68%, 70%, and 67% over L-B, B-ERS, and AODV respectively. Furthermore, the routing overhead improvement was up to 72% over L-B, 63% over B-ERS, and 51% over AODV.

Traffic Locality-Expanding Ring Search

Traffic Locality-Expanding Ring Search (TL-ERS) is an improvement to the existing Expanding Ring Search suggested in [61, 115]. In TL-ERS, the broadcast of a route request covers the source node's neighbourhood region as a first ring searching for the target. If the route discovery in this ring proves unsuccessful, the algorithm then establishes a second ring by doubling the size of that of the first. If the route discovery does not succeed the algorithm

finally resorts to flooding. TL-ERS incurs lower routing overhead compared to that of ERS and AODV with simple flooding without introducing any extra end-to-end delay.

Our simulation results show that TL-ERS exhibits a performance advantage over both ERS and simple flooding used in AODV by improving the end-to-end delay, reducing route request latency, losing fewer packets, and incurring lower routing overhead. For instance, in network size analysis TL-ERS improves end-to-end delay by up to 38% over ERS and up to 44% over AODV. Moreover, TL-ERS reduces routing overhead by up to 42% over ERS and by up to 98% over AODV.

Comparison of the new algorithms

Comparing the simulation results of our new route discovery algorithms reveals the following:

- TLRDA-D and TLRDA-C achieve almost the same low end-to-end delay. However, TLRDA-C incurs lower routing overhead than TLRDA-D. Compared to TL-ERS, they both give lower end-to-end delay but higher routing overhead. These two observations are true for all our performed scenarios. For instance, the end-to-end delay and routing overhead are shown in Table 7-1 when varying network size.

Table 7-1. Improvments of the new algorithms over AODV when varying network size.

Algorithm	End-to-end delay		Routing Overhead	
	Small	Moderate	Small	Moderate
TLRDA-D	52%	67%	0%	-15%
TLRDA-C	53%	68%	17%	51%
TL-ERS	44%	38%	89%	98%

- TLRDA-D and TLRDA-C are likely to be most suitable for time sensitive applications such as instant messaging applications. Furthermore, TLRDA-C is best for applications that are both time and overhead sensitive such as fire fighters working in teams. However, TL-ERS is most suitable for overhead sensitive applications such as groups of college students exchanging email messages.

7.3 Directions for future work

Several interesting issues and open problems that require further investigations have emerged during the course of this research. These are briefly outlined below.

- Most of MANETs research have used simulation to evaluate the performance of the algorithms suggested as in [6, 33, 34, 38, 71, 79]. However, it might not be possible to examine large-scale scenarios using the simulation approach due to time and complexity constraints highlighting the importance of analytical models. Some analytical models have been developed [46, 57, 109, 129] considering some but not all MANETs characteristics. A recent study in [127] concentrated on mobility and lifetime of links for two entity mobility models. Therefore, developing analytical models for MANETs that take into consideration all the important features of MANETs including mobility and power would be desirable as they would allow the investigation of the performance behaviour of these systems under scenarios that might not be possible to consider by means of simulations such as large networks operating under heavy traffic conditions.

- Simulation is an important tool for studying MANETs. However, simulation always requires certain assumptions to keep the model at a manageable level. Consequently, the model may not capture all the factors that might affect the system performance due to those assumptions. Moreover, some important characteristics of MANETs such as energy consumption and radio propagation are inherently hard to model accurately in the simulation models. So far, there has been little work in the literature on the deployment and performance measurement of real practical MANET systems such as [82] due to time and cost limitations.

- In real-life experiments, the whole system is tested in a practical environment. Testing our new algorithms are ease to deploy in a real experiment since they can be implemented as extra functions on top of the on-demand routing protocol without extra cost because it requires no extra hardware. Networks equipped with our algorithms should work better in scalable environment since each source node has its own neighbourhood region to work with and avoid covering the whole network with fulfilled route requests which makes them suitable for energy-constrained networks. Also, such algorithms are kept simple

where each source node only needs to maintain its own *LP* parameter to achieve low complexity. They should work with any compatible existing technology designed for MANETs' environment. Such experiments can validate our simulations findings and help calibrate future simulation models. It would be complementary to our work to conduct such experiments provided that adequate resources are available.

- Synthetic traffic and mobility scenarios have been used in our simulation runs as in most other studies on MANETs [19, 50, 71, 73, 78, 79, 81]. It is important to study the behaviour of the new algorithms using traces collected from real experiments such as [105] where nodes generate random traffic and move according to human-driven approximation of the RWP mobility model. Hopefully, more real traces will become available in the near future as more real MANET experiments are conducted.

- The performance of our new algorithms has been analysed assuming a homogeneous network in a pure ad hoc mode where all nodes are mobile. It would be interesting to investigate their behaviour in heterogeneous networks where MANET is connected to an infrastructure network [117].

- This research has considered the Reference Point Group Mobility (RPGM) model to simulate mobility. It would be interesting to examine the behaviour of our algorithms under different group mobility models such as the Reference Velocity Group Mobility Model (RVGMM) [121] or the Reference Region Group Mobility (RRGM) model [87] depending on the simulated scenarios or any special purpose models such as [48] which intended for social networks.

- For simplicity and predictability, CBR traffic has been used to assess the performance of our algorithms as well as with other algorithms for fair comparison. A natural extension of this work would be to analyse the behaviour of our algorithms under other traffic types such as VBR or under a different transport protocol such as TCP.

- In proactive routing protocols, nodes collect topological information from the periodically exchanged information between each other and maintain them in their routing tables. It would be interesting to explore the possibility of using the traffic locality approach to improve the broadcasting of the periodical information. One possibility would be to broadcast the periodical massages more often within the source node's neighbourhood

region. Also, the traffic locality can be utilised to improve the hybrid routing algorithms such as ZRP [49].

- The performance evaluation has been carried out in the context of AODV routing protocol that uses simple flooding. A natural extension of this work would investigate the performance merits of other on-demand routing algorithms such as DSR [60] and Associativity-Based Routing (ABR) [21].

References

[1] "IEEE standard for Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," IEEE Computer Society, 1999.

[2] M. Abolhasan, T. Wysocki, and E. Dutkiewicz, "A review of routing protocols for mobile ad hoc networks," *Ad Hoc Networks*, vol. 2, pp. 1-22, 2004.

[3] C. Adjih, T. Clausen, P. Jacquet, et al., "Optimized Link State Routing Protocol," The Internet Engineering Task Force, IETF,RFC 3626, 2003.

[4] G. Aggelou, *Mobile Ad Hoc Networks: From Wireless LANs to 4G Networks*, 1 ed, McGraw-Hill Professional, 2004.

[5] I. F. Akyildiz and W. Xudong, "A survey on wireless mesh networks," *Communications Magazine, IEEE*, vol. 43, pp. S23-S30, 2005.

[6] A. Al-Maashri and M. Ould-Khaoua, "Performance Analysis of MANET Routing Protocols in the Presence of Self-Similar Traffic," presented at 31st IEEE Conference on Local Computer Networks, Tampa, FL, 2006.

[7] G. Anastasi, M. Conti, and E. Gregori, "IEEE 802.11 Ad Hoc Networks: Protocols, Performance and Open Issues," in *Mobile Ad hoc networking*, S. Basagni, M. Conti, S. Giordano, et al., Eds. IEEE Press, John Wiley and Sons, Inc., 2003.

[8] P. Andrzej, "Broadcasting in radio networks," in *Handbook of wireless networks and mobile computing*, I. Stojmenovic, Ed. John Wiley & Sons, Inc., pp. 509-528, 2002.

[9] M. Arunesh, Nick L. Petroni, Jr., A. A. William, et al., "Security issues in IEEE 802.11 wireless local area networks: a survey: Research Articles," *Wirel. Commun. Mob. Comput.*, vol. 4, pp. 821-833, 2004.

[10] F. Bai, N. Sadagopan, and A. Helmy, "The IMPORTANT framework for analyzing the Impact of Mobility on Performance Of RouTing protocols for Adhoc NeTworks," *Ad Hoc Networks*, vol. 1, pp. 383-403, 2003.

[11] F. Bai, N. Sadagopan, and A. Helmy, "IMPORTANT: An evaluation framework to study the Impact of Mobility Patterns On RouTing in Ad-hoc NeTworks," University of Southern California, 2005.

[12] F. Bai, N. Sadagopan, B. Krishnamachari, et al., "Modeling path duration distributions in MANETs and their impact on reactive routing protocols," *IEEE Journal on Selected Areas in Communications*, vol. 22, pp. 1357-1373, 2004.

[13] D. Bertsekas and R. Gallager, *Data Networks*, Second ed, Prentice Hall, 1992.

[14] J. Billington, D. Michel, and R. Grzegorz, *Application of Petri Nets to Communication Networks, Advances in Petri Nets*, vol. 1605, Springer-Verlag, 1999.

[15] R. Bindal, P. Cao, W. Chan, et al., "Improving Traffic Locality in BitTorrent via Biased Neighbor Selection," presented at 26th IEEE International Conference on Distributed Computing Systems (ICDCS'06), Lisboa, Portugal, 2006.

[16] G. V. Bochmann, "Finite state description of communication protocols," *Computer Networks*, vol. 2, pp. 361-372, 1978.

[17] F. Borgonovo, "ExpressMAN: Exploiting Traffic Locality in Expressnet," *IEEE Journal on Selected Areas in Communications*, vol. 5, pp. 1436-1443, 1987.

[18] J. Broch, A. M. David, B. J. David, et al., "A performance comparison of multi-hop wireless ad hoc network routing protocols," presented at 4th annual ACM/IEEE international conference on Mobile computing and networking, Dallas, Texas, United States, 1998.

[19] T. Camp, J. Boleng, and V. Davies, "A Survey of Mobility Models for Ad Hoc Network Research," *Wireless Communications & Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications*, vol. 2, pp. 483-502, 2002.

[20] R. Castaeda, S. Das, and M. Marina, "Query localization techniques for on-demand routing protocols in ad hoc networks," *Wireless Networks*, vol. 8, pp. 137-151, 2002.

[21] T. Chai-Keong, "Associativity-Based Routing for Ad Hoc Mobile Networks," *Wirel. Pers. Commun.*, vol. 4, pp. 103-139, 1997.

[22] I. D. Chakeres and C. E. Perkins, "Dynamic MANET On-demand Routing Protocol," IETF Internet Draft, 2008.

[23] N. Chang and M. Liu, "Revisiting the TTL-based controlled flooding search: optimality and randomization," presented at 10th annual international conference on Mobile computing and networking, Philadelphia, PA, USA, 2004.

[24] Z. Cheng and W. Heinzelman, "Flooding strategy for target discovery in wireless networks," presented at 6th ACM international workshop on Modeling analysis and simulation of wireless and mobile systems, San Diego, CA, USA, 2003.

[25] I. Chlamtac, M. Conti, and J. J. N. Liu, "Mobile ad hoc networking: imperatives and challenges," *Ad Hoc Networks*, vol. 1, pp. 13-64, 2003.

[26] E. J. Christine, M. S. Krishna, A. Prathima, et al., "A Survey of Energy Efficient Network Protocols for Wireless Networks," *Wirel. Netw.*, vol. 7, pp. 343-358, 2001.

[27] T. Clausen, C. Dearlove, and B. Adamson, "Jitter Considerations in Mobile Ad Hoc Networks (MANETs)," The Internet Engineering Task Force, IETF, RFC 5148, 2008.

[28] M. A. Cohen and A. J. Cohen, "Optimized Network Engineering Tools (OPNET)," Bethesda, Maryland, USA, http://www.opnet.com/, 1986, [May 2009].

[29] C. Cordeiro and D. P. Agrawal, *Ad Hoc & Sensor Networks: Theory And Applications*, Illustrated ed, World Scientific Publishing Company, 2006.

[30] F. Dai and J. Wu, "Performance Analysis of Broadcast Protocols in Ad Hoc Networks Based on Self-Pruning," *IEEE Trans. Parallel Distrib. Syst.*, vol. 15, pp. 1027-1040, 2004.

[31] W. Dally and B. Towles, *Principles and Practices of Interconnection Networks* Illustrated ed, Morgan Kaufmann Publishers, 2004.

[32] S. R. Das, R. Castaneda, Y. Jiangtao, et al., "Comparative performance evaluation of routing protocols for mobile ad hoc networks," presented at The 7th IEEE International Conference on Computer Communications and Networks (IC3N'98), Lafayette, Louisiana, 1998.

[33] S. R. Das, C. Robert, eda, et al., "Simulation-based performance evaluation of routing protocols for mobile ad hoc networks," *ACM/Baltzer Mobile Networks and Applications (MONET) Journal*, vol. 5, pp. 179-189, 2000.

[34] C. David, S. Yoav, Andr, et al., "On the accuracy of MANET simulators," presented at second ACM international workshop on Principles of mobile computing, Toulouse, France, 2002.

[35] V. Davies, "Evaluating Mobility Models within an Ad Hoc Network," Colorado School of Mines, M.S. Thesis, 2000.

[36] P. Denning, "The locality principle," *Commun. ACM*, vol. 48, pp. 19-24, 2005.

[37] P. Denning, "The working set model for program behavior," *Commun. ACM*, vol. 11, pp. 323-333, 1968.

[38] H. Dubois-Ferriere, M. Grossglauser, and M. Vetterli, "Age matters: efficient route discovery in mobile ad hoc networks using encounter ages," presented at 4th ACM international symposium on Mobile ad hoc networking and computing, Annapolis, Maryland, USA, 2003.

[39] J. Eisbrener, G. Murphy, D. Eade, et al., "Recycled path routing in mobile ad hoc networks," *Computer Communications*, vol. 29, pp. 1552-1560, 2006.

[40] I. Eltahir, "The Impact of Different Radio Propagation Models for Mobile Ad hoc NETworks (MANET) in Urban Area Environment," in *2nd International Conference on Wireless Broadband and Ultra Wideband Communications*. Sydney, Australia,: IEEE Computer Society, pp., 2007.

[41] K. Fall, "NS Notes and Documentation," http://www.isi.edu/nsnam/ns/ns-documentation.html, 2000, [May 2009].

[42] P. J. Fortier and H. E. Michel, *Computer Systems Performance Evaluation and Prediction*, 1st ed, Digital Press, 2002.

[43] R. Fujimoto, M. , K. Perumalla, and G. Riley, *Network Simulation (Synthesis Lectures on Communication Networks)*, Morgan and Claypool Publishers, 2007.

[44] L. Gargano, M. Hammar, and A. Pagh, "Limiting flooding expenses in on-demand source-initiated protocols for mobile wireless networks," presented at

18th International Parallel and Distributed Processing Symposium, Santa Fe, New Mexico, 2004.

[45] S. Giordano, "Mobile Ad-Hoc Networks," in *Handbook of Wireless Networks and Mobile Computing*, I. Stojmenovic, Ed. John Wiley & Sons, 2002.

[46] R. Groenevelt, P. Nain, and G. Koole, "The message delay in mobile ad hoc networks," *Performance Evaluation*, vol. 62, pp. 210-228, 2005.

[47] N. Gulati, "LAN Traffic Locality: Characterization and Application," *Department of Computational Science*, University of Saskatchewan, Saskatoon, Saskatchewan, M.S. Thesis, 1992.

[48] S. Gunasekaran and N. Nagarajan, "A new group mobility model for mobile adhoc network based on unified relationship matrix," *WTOC*, vol. 7, pp. 58-67, 2008.

[49] Z. J. Haas, M. R. Pearlman, and P. Samar, "The Zone Routing Protocol (ZRP) for Ad Hoc Networks," IETF MANET Working Group, INTERNET-DRAFT July, 2002.

[50] A. Hamidian, "A Study of Internet Connectivity for Mobile Ad Hoc Networks in NS 2," Lund Institute of Technology, Sweden, M.S. Thesis, 2003.

[51] J. Hassan and S. Jha, "On the optimization trade-offs of expanding ring search," *Springer LNCS*, vol. 3326, pp. 489-494, 2004.

[52] M. Hassan and R. Jain, *High Performance TCP/IP Networking: Concepts, Issues, and Solutions*, Prentice-Hall, 2003.

[53] X. Hong, G. Mario, P. Guangyu, et al., "A group mobility model for ad hoc wireless networks," presented at 2nd ACM international workshop on Modeling, analysis and simulation of wireless and mobile systems, Seattle, Washington, United States, 1999.

[54] Z. Hongqiang, W. Jianfeng, C. Xiang, et al., "Medium access control in mobile ad hoc networks: challenges and solutions: Research Articles," *Wirel. Commun. Mob. Comput.*, vol. 6, pp. 151-170, 2006.

[55] IEEE, "Standard for Information technology Telecommunications and information exchange between systems Local and metropolitan area networks

Specific requirements, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. IEEE Computer Society," 2007.

[56] M. Ilyas and R. C. Dorf, *The handbook of ad hoc wireless networks*, CRC Press, Inc., 2003.

[57] P. Jacquet, A. M. Naimi, and G. Rodolakis, "Routing on asymptotic delays in IEEE 802.11 wireless ad hoc networks " presented at 1st workshop on Resource Allocation in Wireless NETworks, in conjunction with IEEE WiOpt 2005, Riva Del Garda, Italy, 2005.

[58] L. Jinyang, B. Charles, S. J. D. C. Douglas, et al., "Capacity of Ad Hoc wireless networks," presented at 7th annual international conference on Mobile computing and networking, Rome, Italy, 2001.

[59] M. Joa-Ng and I.-T. Lu, "A peer-to-peer zone-based two-level link state routing for mobile ad hoc networks," *IEEE Journal on Selected Areas in Communications*, vol. 17, pp. 1415-1425, 1999.

[60] D. Johnson, D. Maltz, and Y.-C. Hu, "The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR)," The Internet Engineering Task Force, IETF, draft-ietf-manet-dsr-09.txt, April 2003.

[61] D. B. Johnson and D. A. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Networks," in *Mobile Computing*, vol. 353, I. a. Korth, Ed. Kluwer Academic Publishers, pp. 153-181, 1996.

[62] O. Katia, V. Kumar, and T. Gene, "Flooding for reliable multicast in multi-hop ad hoc networks," *Wireless Networks*, vol. 7, pp. 627-634, 2001.

[63] L. Kleinrock, *Queueing Systems, Volume 2: Computer Applications.*, John Wiley & Sons, Inc., 1976.

[64] D. Koutsonikolas, S. M. Das, H. Pucha, et al., "On optimal TTL sequence-based route discovery in MANETs," presented at the 2nd International Workshop on Wireless Ad Hoc Networking, Columbus, Ohio, USA, 2005.

[65] C. Kozierok, *The TCP/IP Guide*, 1st ed, No Starch Publishing, 2005.

[66] W. Kun, W. Meng, W. Lu, et al., "A Novel Location-Aided Routing Algorithm for MANETs," presented at Fifth International Conference on Information Technology: New Generations (ITNG '08), Las Vegas, Nevada, USA, 2008.

[67] S. Kurkowski, T. Camp, and M. Colagrosso, "MANET simulation studies: the incredibles," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 9, pp. 50-61, 2005.

[68] J. F. Kurose and K. W. Ross, *Computer Networking: A Top-Down Approach* 4th ed, Addison Wesley, 2007.

[69] R. Leung, L. Jilei, E. Poon, et al., "MP-DSR: a QoS-aware multi-path dynamic source routing protocol for wireless ad-hoc networks," presented at 26th Annual IEEE Conference on Local Computer Networks (LCN 2001), Tampa, FL, USA, 2001.

[70] J. Li and M. Prasant, "A novel mechanism for flooding based route discovery in ad hoc networks," presented at IEEE Global Telecommunications Conference (GLOBECOM '03), San Francisco, CA, USA, 2003.

[71] J. Li and M. Prasant, "PANDA: A novel mechanism for flooding based route discovery in ad hoc networks," *Wireless Networks*, vol. 12, pp. 771-787, 2006.

[72] H. Lim and C. Kim, "Flooding in wireless ad hoc networks," *Computer Communications*, vol. 24, pp. 353-363, 2001.

[73] Y. Lu, Y. Zhong, and B. Bhargava, "Packet Loss in Mobile Ad Hoc Networks," Purdue University, USA, April 01, 2003.

[74] J. Luo, P. T. Eugster, and J.-P. Hubaux, "PILOT: Probabilistic Lightweight grOup communication sysTem for Mobile Ad Hoc Networks," *IEEE Transactions on Mobile Computing*, vol. 3, pp. 164-179, 2004.

[75] M. K. Marina and S. R. Das, "On-demand multipath distance vector routing in ad hoc networks," presented at 9th IEEE International Conference on Network Protocols (ICNP'01), Riverside, CA, USA, 2001.

[76] C. McDonald, "The cnet network simulator (v3.1.3)," The University of Western Australia, http://www.csse.uwa.edu.au/cnet3/, 1991, [May 2009].

[77] N. Migas, W. J. Buchanan, and K. A. McArtney, "Mobile agents for routing, topology discovery and automatic network reconfiguration in ad-hoc networks,"

presented at 10th IEEE International Conference and Workshop on the Engineering of Computer-Based Systems (ECBS'03), Huntsville, Alabama, USA, 2003.

[78] M. Minematsu, M. Saito, H. Aida, et al., "HOWL: an efficient route discovery scheme using routing history in ad hoc networks," presented at 27th Annual IEEE International Conference on Local Computer Networks (LCN'02), Tampa, FL, USA, 2002.

[79] I. Ming-Yee, "Selective Flooding in Ad Hoc Networks," *Computer Science Department*, University of Waterloo, Ontario, Canada, M.S. Thesis, 2002.

[80] N. Moghim, F. Hendessi, and N. Movehhedinia, "An improvement on ad-hoc wireless network routing based on AODV," presented at 8th International Conference on Communication Systems, ICCS, 2002.

[81] M. Mosko and J. Garcia-Luna-Aceves, "Performance of group communication over ad hoc networks," presented at IEEE Int. Symp. Computers and Communications ISCC, Italy, 2002.

[82] G. Mulas, A. La Piana, F. Cau, et al., "A multi-hop MANET demonstrator tested on real-time applications," presented at IEEE International Conference on Wireless And Mobile Computing, Networking And Communications (WiMob'05), Montreal (QC), Canada, 2005.

[83] A. Muqattash, M. Krunz, and W. E. Ryan, "Solving the near-far problem in CDMA-based ad hoc networks," *Ad Hoc Networks*, vol. 1, pp. 435-453, 2003.

[84] S. Murthy and B. Manoj, *Ad Hoc Wireless Networks: Architectures and Protocols*, Prentice Hall, 2004.

[85] M. Nauman and F. Muddassar, "Vulnerability analysis and security framework (BeeSec) for nature inspired MANET routing protocols," presented at 9th annual conference on Genetic and evolutionary computation, London, England, 2007.

[86] W. Navidi and T. Camp, "Stationary distributions for the random waypoint mobility model," *IEEE Transactions on Mobile Computing*, vol. 3, pp. 99-108, 2004.

[87] J. M. Ng and Y. Zhang, "Reference region group mobility model for ad hoc networks," presented at Second IFIP International Conference on Wireless and Optical Communications Networks, WOCN, 2005.

[88] B. O'Hara and A. Petrick, *IEEE 802.11 Handbook: A Designer's Companion*, Second ed, The Institute of Electrical and Electronics Engineers, 2005.

[89] I. Park, J. Kim, and I. Pu, "Blocking Expanding Ring Search Algorithm for Efficient Energy Consumption in Mobile Ad Hoc Networks," presented at Third Annual Conference on Wireless On-demand Network Systems and Services, Les Menuires, France, 2006.

[90] I. Park and I. Pu, "Energy Efficient Expanding Ring Search," presented at First Asia International Conference on Modelling & Simulation, 2007.

[91] T. K. Paul and T. Ogunfunmi, "Wireless LAN Comes of Age: Understanding the IEEE 802.11n Amendment," *Circuits and Systems Magazine, IEEE*, vol. 8, pp. 28-54, 2008.

[92] W. Peng and L. Xi-Cheng, "On the reduction of broadcast redundancy in mobile ad hoc networks," presented at 1st ACM international symposium on Mobile ad hoc networking and computing, Boston, Massachusetts, 2000.

[93] C. Perkins, *Ad hoc networking*, Addison Wesley, 2001.

[94] C. Perkins, E. Belding-Royer, and S. Das, "AODV Ad Hoc On-Demand Distance Vector Routing," The Internet Engineering Task Force, IETF, RFC 3561, July, 2003.

[95] C. E. Perkins and P. Bhagwat, "Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers," presented at SIGCOMM '94 Conference on Communications Architectures, Protocols and Applications, London, UK, 1994.

[96] L. F. Perrone, Y. Yuan, and D. M., "Modeling and simulation best practices for wireless ad hoc networks," presented at Winter Simulation Conference, New Orleans, Louisiana, USA, 2003.

[97] J. L. Peterson, "Petri Nets," *ACM Computing Surveys (CSUR)*, vol. 9, pp. 223-252, 1977.

[98] M. Prasant and K. Srikanth, *AD HOC NETWORKS: Technologies and Protocols*, Springer-Verlag New York, Inc., 2004.

[99] H. Pucha, S. M. Das, and Y. C. Hu, "The performance impact of traffic patterns on routing protocols in mobile ad hoc networks," *Computer Networks*, vol. 51, pp. 3595-3616, 2007.

[100] A. Qayyum, L. Viennot, and A. Laouiti, "Multipoint Relaying for Flooding Broadcast Messages in Mobile Wireless Networks," presented at 35th Annual Hawaii International Conference on System Sciences (HICSS'02)-Volume 9, 2002.

[101] Z. Qi and P. A. Dharma, "Dynamic probabilistic broadcasting in MANETs," *J. Parallel Distrib. Comput.*, vol. 65, pp. 220-233, 2005.

[102] G. Rajiv, P. Srinivasan, and M. Arunesh, "Minimizing broadcast latency and redundancy in ad hoc networks," presented at 4th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC), Annapolis, Maryland, USA, 2003.

[103] R. Rajmohan, "Topology control and routing in ad hoc networks: a survey," *SIGACT News*, vol. 33, pp. 60-73, 2002.

[104] S. Ramanathan and S. Martha, "A survey of routing techniques for mobile communications networks," *Mob. Netw. Appl.*, vol. 1, pp. 89-104, 1996.

[105] S. G. Robert, K. David, N. Calvin, et al., "Outdoor experimental comparison of four ad hoc routing algorithms," Downloadable from http://crawdad.cs.dartmouth.edu/dartmouth/outdoor/routing, 2006, [May 2009].

[106] S. Ross, *Introduction to Probability and Statistics for Engineers and Scientists*, Second ed, Academic Press, 2000.

[107] C. Ruay-Shiung and T. Chang-Zhou, "Adding sense of spatial locality to routing protocols for mobile ad hoc networks: Research Articles," *Wirel. Commun. Mob. Comput.*, vol. 7, pp. 299-310, 2007.

[108] Y. Shaohu, Y. Shaohu, Z. Yongning, et al., "Priority backoff algorithm for IEEE 802.11 DCF," presented at International Conference on Communications, Circuits and Systems (ICCCAS '04), 2004.

[109] T. Sheltami and H. Mouftah, "Clusterhead controlled token for virtual base station on-demand in MANETs," presented at 23rd International Conference on Distributed Computing Systems Workshops, 2003.

[110] W. Shih-Hsien, C. Ming-Chieh, and H. Ting-Chao, "Zone-based controlled flooding in mobile ad hoc networks," presented at International Conference on Wireless Networks, Communications and Mobile Computing, 2005.

[111] M. Shikharesh and B. B. Richard, "Measurement and analysis of locality phases in file referencing behaviour," presented at ACM SIGMETRICS joint international conference on Computer performance modelling, measurement and evaluation, Raleigh, North Carolina, United States, 1986.

[112] A. Silberschatz, P. Galvin, and G. Gagne, *Operating Systems Concepts*, 7th ed, John Wiley & Sons, 2005.

[113] M. Spohn and J. J. Garcia-Luna-Aceves, "Improving route discovery in on-demand routing protocols using two-hop connected dominating sets," *Ad Hoc Networks*, vol. 4, pp. 509-531, 2006.

[114] W. Stallings, "IEEE 802.11:Wireless LANs from a to n," *IT Professional*, vol. 6, pp. 32-37, 2004.

[115] L. Sung-Ju, E. Belding-Royer, and C. Perkins, "Scalability study of the ad hoc on-demand distance vector routing protocol," *Int. J. Netw. Manag.*, vol. 13, pp. 97-114, 2003.

[116] N. Sze-Yao, T. Yu-Chee, C. Yuh-Shyan, et al., "The broadcast storm problem in a mobile ad hoc network," presented at 5th annual ACM/IEEE international conference on Mobile computing and networking, Seattle, Washington, United States, 1999.

[117] A. Tanenbaum, *Computer Networks*, 4th ed, Pearson Education, 2003.

[118] A. Varga and H. Rudolf, "An overview of the OMNeT++ simulation environment," presented at Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops, Marseille, France, 2008.

[119] E. Vollset and P. Ezhilchelvan, "A Survey of Reliable Broadcast Protocols for Mobile Ad-hoc Networks," University of Newcastle upon Tyne, 2003.

[120] L. Walters and P. S. Kritzinger, "Cellular networks: past, present and future," *Crossroads*, vol. 7, pp. 4-18, 2000.

[121] K. H. Wang and B. Li, "Group mobility and partition prediction in wireless ad-hoc networks," presented at IEEE International Conference on Communications (ICC 2002), New York, USA, 2002.

[122] B. Williams and C. Tracy, "Comparison of broadcasting techniques for mobile ad hoc networks," presented at 3rd ACM international symposium on Mobile ad hoc networking and computing, Lausanne, Switzerland, 2002.

[123] J. Wu and D. Fei, "Efficient Broadcasting with Guaranteed Coverage in Mobile Ad Hoc Networks," *IEEE Transactions on Mobile Computing*, vol. 4, pp. 259-270, 2005.

[124] J. Wu and D. Fei, "A Generic Broadcast Protocol in Ad Hoc Networks Based on Self-Pruning," presented at 17th International Symposium on Parallel and Distributed Processing, Nice, France, 2003.

[125] K. Wu, C. Huang, and F. Li, "Pessimism is mostly the best in the expanding ring search for wireless networks," presented at IEEE Global Telecommunications Conference (Globecom'06), San Francisco, California, 2006.

[126] T. Wu, C. Huang, and H. Chao, "A survey of Mobile IP in cellular and Mobile Ad-Hoc Network environments," *Ad Hoc Networks*, vol. 3, pp. 351-370, 2005.

[127] X. Wu, H. R. Sadjadpour, J. J. Garcia-Luna-Aceves, et al., "A hybrid view of mobility in MANETs: Analytical models and simulation study," *Computer Communications*, vol. 31, pp. 3810-3821, 2008.

[128] Z. Xiang, B. Rajive, and G. Mario, "GloMoSim: a library for parallel simulation of large-scale wireless networks," *SIGSIM Simul. Dig.*, vol. 28, pp. 154-161, 1998.

[129] L. Xin, Q. Zhen, H. Bo, et al., "Theoretical Analysis on the Performance of Two Broadcast Schemes in Mobile Ad Hoc Networks," presented at 6th International Conference on ITS Telecommunications, 2006.

[130] K. Young-Bae and H. V. Nitin, "Location-aided routing (LAR) in mobile ad hoc networks," *Wireless Networks*, vol. 6, pp. 307-321, 2000.

[131] T. Yu-Chee, N. Sze-Yao, C. Yuh-Shyan, et al., "The broadcast storm problem in a mobile ad hoc network," *Wireless Networks*, vol. 8, pp. 153-167, 2002.

[132] H. Zhang and Z. P. Jiang, "On reducing broadcast expenses in ad hoc route discovery," presented at the 2nd International Workshop on Wireless Ad Hoc Networking (WWAN), Columbus, Ohio, USA, 2005.

Publications during the course of this research

Scientific Journals:

- M. Al-Rodhaan, L. M. Mackenzie, and M. Ould-Khaoua, "Traffic Locality Oriented Route Discovery Algorithm with Delay," Accepted for publication in the International Journal of Computers and Their Applications (IJCA), December, 2009.
- M. Al-Rodhaan, L. M. Mackenzie, and M. Ould-Khaoua, "On the Performance of Traffic Locality Oriented Route Discovery Algorithm with Delay," Accepted for publication in the International Journal of Communications, Network and System Sciences (IJCNS), Scientific Research Publishing, California, USA, 2009.
- M. Al-Rodhaan, L. M. Mackenzie, and M. Ould-Khaoua, "On the Performance of Traffic Locality Oriented Route Discovery Algorithm with Chase Packets," Bahria University Journal of Information & Communication Technology (BUJICT), Vol. 1, December, 2008.
- M. Al-Rodhaan, L. M. Mackenzie, and M. Ould-Khaoua, "*A New Route Discovery Algorithm for MANETs with Chase Packets*," International Journal of Simulation Systems, Science & Technology Special Issue on: Performance Modelling of Computer Networks, Systems and Services, Vol. 8, no.3, (Invited paper from UKPEW'06), pp 1-12, 2007.
- M. Al-Rodhaan, L. M. Mackenzie, and M. Ould-Khaoua, "*A Traffic Locality Oriented Route Discovery Algorithm for MANETs*," Ubiquitous Computing and Communication Journal (UBICC), Vol. 2, no. 5, pp 58-68, 2007.

Conference Papers:

- M. Al-Rodhaan, L. M. Mackenzie, and M. Ould-Khaoua, "*Traffic Locality Oriented*

Route Discovery Algorithm with Chase Packets for Mobile Ad Hoc Networks," Proceedings of 2009 International Conference on Wireless Networks (ICWN'09), Monte Carlo Resort ,Las Vegas, Nevada, USA, 13-16 July, 2009.

- M. Al-Rodhaan, L. M. Mackenzie, and M. Ould-Khaoua, "*TL-ERS: A Traffic Locality Based Expanding Ring Search for MANETs*," The 8th Annual PostGraduate Symposium on The Convergence of Telecommunications, Networking and Broadcasting (PGNet 2007), Liverpool John Moores University, UK, pp 104-110, 28-29 June 2007.

- M. Al-Rodhaan, L. M. Mackenzie, and M. Ould-Khaoua, "*A new route discovery algorithm for MANETs with chase packets*," 22nd Annual UK Performance Engineering Workshop, UKPEW'06, Bournemouth University, Poole, Dorset, pp 1-8, July 2006.

Technical Reports:

- M. Al-Rodhaan, L. M. Mackenzie, and M. Ould-Khaoua, "*Improvement to Blocking Expanding Ring Search for MANETs,*" DCS Technical Report Series, Dept of Computing Science, University of Glasgow, 2008.

Appendix A: Blocking-ERS Plus

A.1 Introduction

In the presence of mobility, B-ERS suffers from performance degradation, as the simulation analysis in section 5.4 reveals clearly, due to the immature discard of chase packets where most of the time the fulfilled route request manages to escape with the help of mobility from its associated chase packet. B-ERS was explained in detail in Section 5.1.2. In this appendix, we are proposing a new algorithm, Blocking-ERS Plus, to overcome this deficiency in B-ERS. It works by continuing to broadcast chase packets until the catching is insured to maximise the success rate of the catching mechanism.

A.2 Blocking-ERS Plus Algorithm

Blocking-ERS Plus (B-ERS+ for short) is an improvement of B-ERS to increase the success rate which improves network performance in terms of latency and overhead for MANETs. These two algorithms differ only in the processing of the chase packets. In B-ERS+, the chase packet is broadcast beyond the ring where the finder of the route reside as illustrated in Figure A-1 in an effort to catch the fulfilled route request in case intermediate nodes move away from their ring after receiving the route request.

Unlike B-ERS, B-ERS+ does not need to extend the format of the route reply packet because the source node broadcast the chase packet without restricting it to cover only the ring where the finder of the route reside.

```
Steps performed by each node upon receiving the chase packet in B-ERS+
If the chase packet is a duplicate then
    Discard it.
Else
    Store chase information
    If the route request received then
        If the route request broadcasted then
            Broadcast the chase packet.
        Else
            Discard both packets.
        End if
    Else
        Discard the chase packet.
    End if
End if
```

Figure A-1: Steps performed by intermediate nodes upon receiving a chase packet in B-ERS+.

A.3 Simulation

Simulations have been conducted to evaluate B-ERS+ and compare it with TLRDA-C, B-ERS, and simple flooding used in AODV algorithms using ns2 simulator version 2.29 [41]. B-ERS+ was implemented as a modification to the existing AODV implementation. The same case is true for TLRDA-C and B-ERS.

The comparison metrics include the network coverage, end-to-end delay, average route request latency, routing overhead, and packet loss to study the success rate, network latency, network overhead, and congestion level. The simulation analysis considers all the three cases: effect of network size, effect of traffic load, and effect of mobility as stated earlier in the second chapter, Table 2-3.

A.3.1 Effect of network size

Figures A-2 to A-6 display the results of running our algorithm, TLRDA-C and B-ERS+ against both B-ERS and AODV for 900 seconds using networks with different number of nodes, from 20 to 100 in an area of 1000m x 1000m with a minimum speed of 1m/s and a maximum speed of 15m/s. The number of communication sessions is ten.

Figure A-2 shows that the success rate of the catching process improved dramatically for B-ERS+

compared to B-ERS. In B-ERS, the coverage is nearly equal to AODV. While in B-ERS+, it is improved by 76% to 80% compared to AODV. B-ERS+ improvement in terms of success rate over the original B-ERS is 74% to 78% while the success rates in B-ERS+ and TLRDA-C are ranging between -36% and 34%. In some situation B-ERS+ achieves better success rate than TLRDA-C and vice versa; the reason behind the success in B-ERS+ is adding larger amount of delays to route requests than TLRDA-C which might increase end-to-end delay.

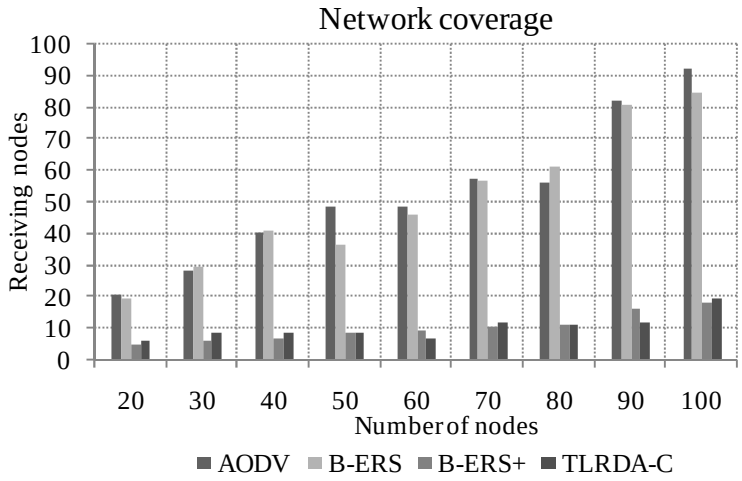

Figure A-2: Network coverage versus network size with 10 communication sessions and 15m/s as maximum speed.

Latency:

Figure A-3 explores the end-to-end delay for TLRDA-C, B-ERS+, B-ERS, and AODV. B-ERS+ reduces the average end-to-end delay more than B-ERS, and AODV because the network in B-ERS+ is less congested. TLRDA-C achieves lower end-to-end delay than B-ERS+ due to the faster propagation of the route request within its neighbourhood region remembering that TLRDA-C broadcasts with less contention as in TLRDA-D. The reason behind the end-to-end delay increment in both B-ERS and B-ERS+ is delaying route requests from start and before discovering the required route. The average end-to-end delay improvement in TLRDA-C is better than B-ERS+ by 59% to 67% while B-ERS+ improves the end-to-end delay by up to 25% over B-ERS and by up to 16% over AODV.

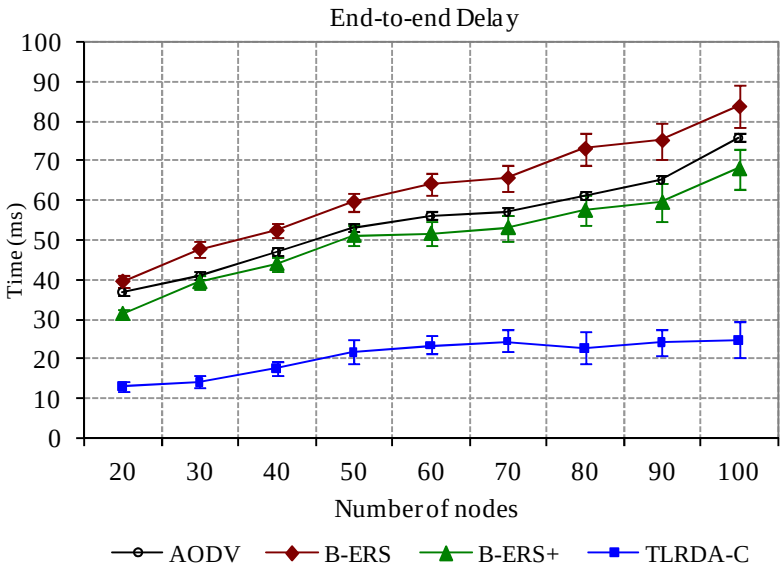

Figure A-3: End-to-end delay versus network size with 10 communication sessions and 15m/s as maximum speed.

Figure A-4 shows the superiority of TLRDA-C by minimising the average of route request latency. The average route request latency of B-ERS+ is reduced more than B-ERS which means that the catching process was more successful in B-ERS+. TLRDA-C improves the average of route request latency by 31% to 62% over B-ERS+ while B-ERS+ improves it by 44% to 60% over B-ERS.

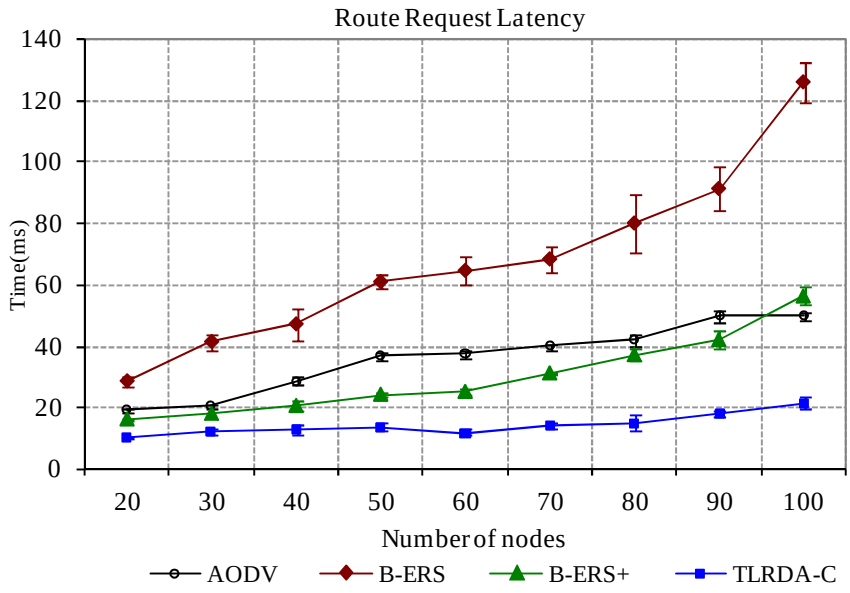

Figure A-4: Route request latency versus network size with 10 sessions and 15m/s as maximum speed.

Overhead:

Figure A-5 depicts the routing overhead for all four algorithms. B-ERS+ reduces the number of received route request but increases the number of chase packets received compared to B-ERS. Nevertheless, the routing overhead in B-ERS+ is improved by 45% to 55% over B-ERS and by 33% to 40% over AODV. This improvement increases with the increment of network size. TLRDA-C reduces routing overhead more than B-ERS+ in moderate size networks by 28% while B-ERS+ reduces it more in small size environment by 24%.

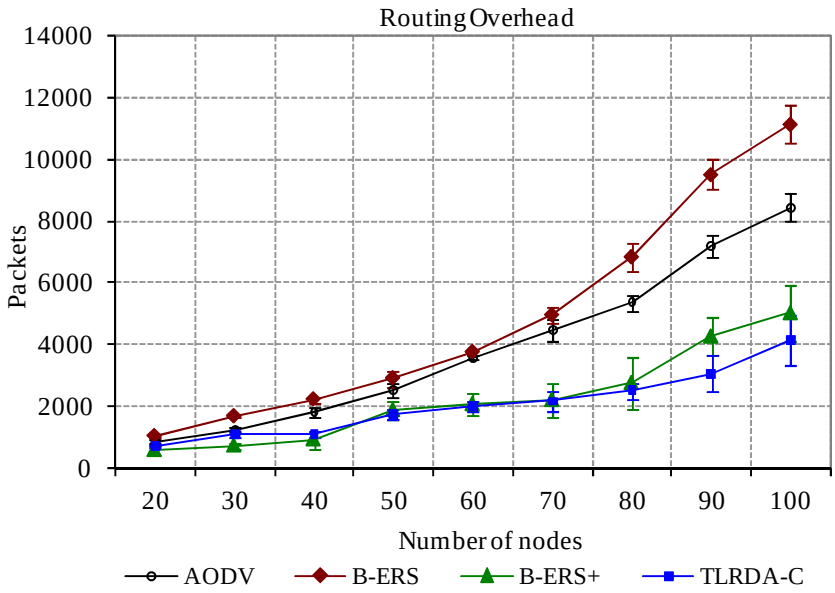

Figure A-5: Routing overhead versus network size with 10 sessions and 15m/s as maximum speed.

Congestion:

Figure A-6 shows the packet loss for all four algorithms. B-ERS+ reduces the packet loss of B-ERS by 22% to 67% and by 23% to 58% over AODV. B-ERS+ improvement increases with the increment of network size. TLRDA-C reduces packet loss more than B-ERS+ in low moderate (70 nodes) to moderate size networks by up to 13% while B-ERS+ reduces it more in small to low moderate (60 nodes) network environment by up to 12%.

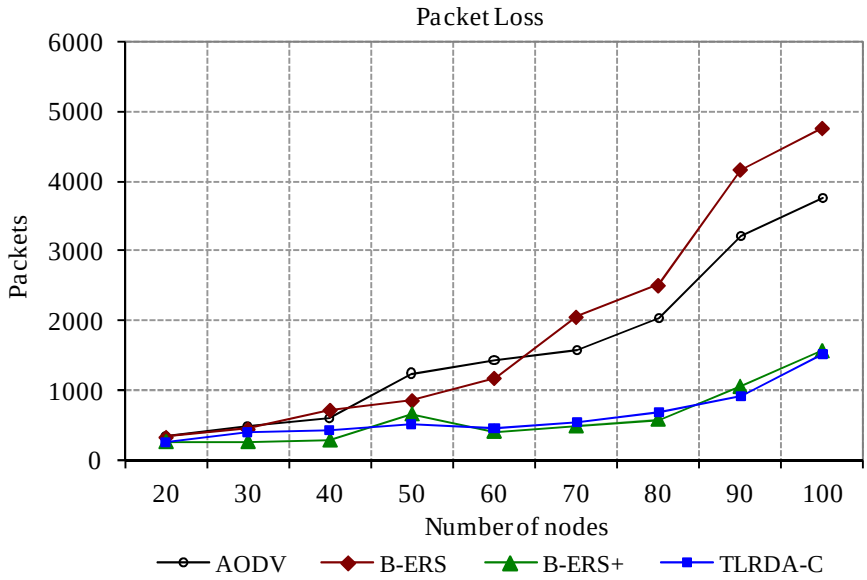

Figure A-6: Packet loss versus network size with 10 sessions and 15m/s as maximum speed.

Therefore, the network performance is improved for B-ERS+ compared to B-ERS and AODV by reducing latency and overhead due to the higher success rate of the catching process in B-ERS+. TLRDA-C improves the network performance more in terms of latency compared to B-ERS+. This improvement increases with moderate size networks.

A.3.2 Effect of traffic load

Figures A-7 to A-11 display the results of running TLRDA-C and B-ERS+ against AODV and B-ERS for 900 seconds using networks of size 70 nodes in an area of 1000m x 1000m with a random speed ranging between 1m/s and 15m/s. The amount of traffic ranges from 5 to 35 communication sessions incremented by five.

Figure A-7 demonstrates how much the network is covered. B-ERS+ improves the success rate of B-ERS dramatically by 85% and 87%. B-ERS+ catches more route requests than TLRDA-C by up to 32% because it imposes larger amount of delays to route request which enables the chase packet to reach the associated route request earlier.

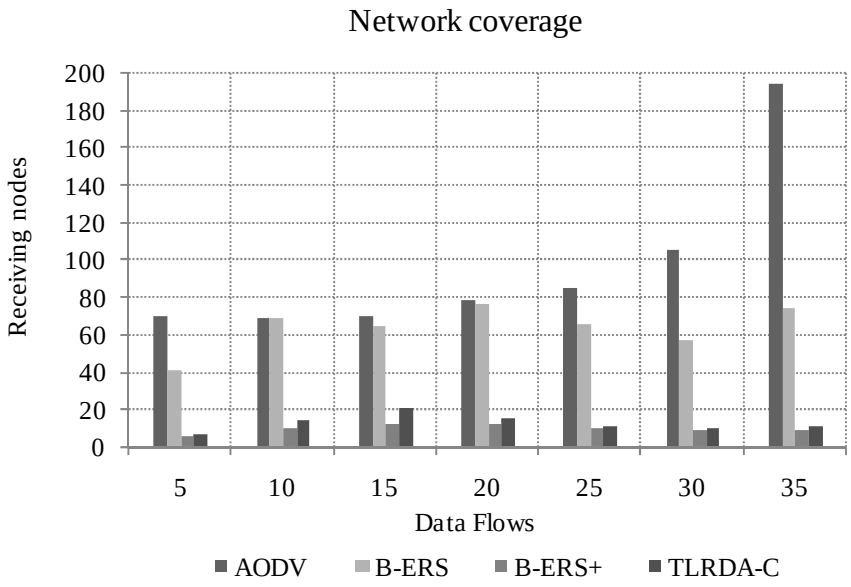

Figure A-7: Network coverage versus traffic load in a network of 70 nodes and 15m/s as maximum speed.

Latency:

Figure A-8 shows that B-ERS+ improves the end-to-end delay over B-ERS because B-ERS+ frees the network from unneeded route requests which reduces the network congestion. This improvement ranges from 39% to 44% over B-ERS and 26% to 54% over AODV. B-ERS+ still suffers from high end-to-end delay due to imposing delay to route request propagation before discovering the route. TLRDA-C achieves end-to-end delay better than B-ERS+ by 31% to 40%.

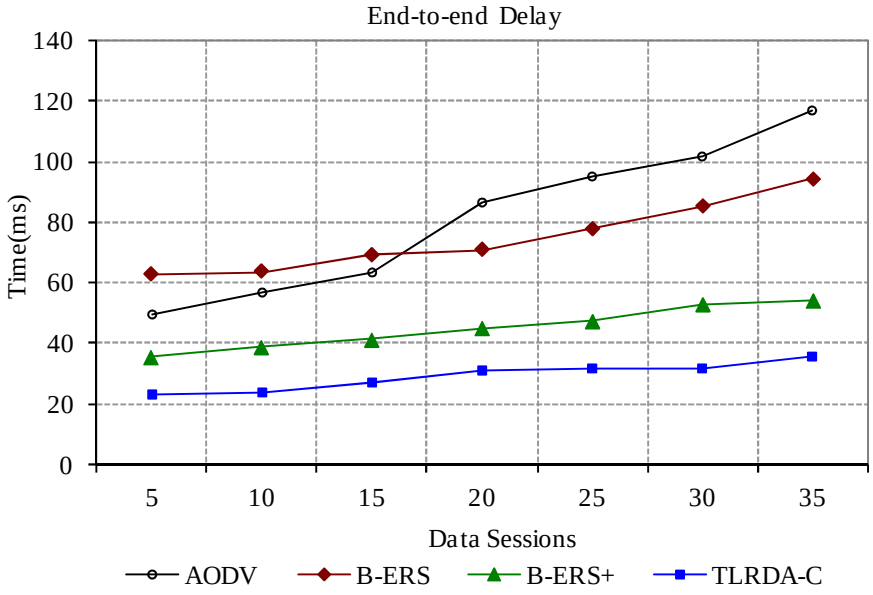

Figure A-8: End-to-end delay versus traffic load in a network of 70 nodes and 15m/s as maximum speed.

Figure A-9 reveals the superiority of TLRDA-C among all four algorithms in terms of the average of route request latency because of the higher success rate in the catching process. The route request latency increases with traffic load due to the increment in the number of packets in the network which adds more contention and may result in more collision. TLRDA-C improves the average of route request latency by 42% to 55% over B-ERS+ while B-ERS+ improves route request latency over B-ERS by 30% to 42% and 17% to 24% over AODV.

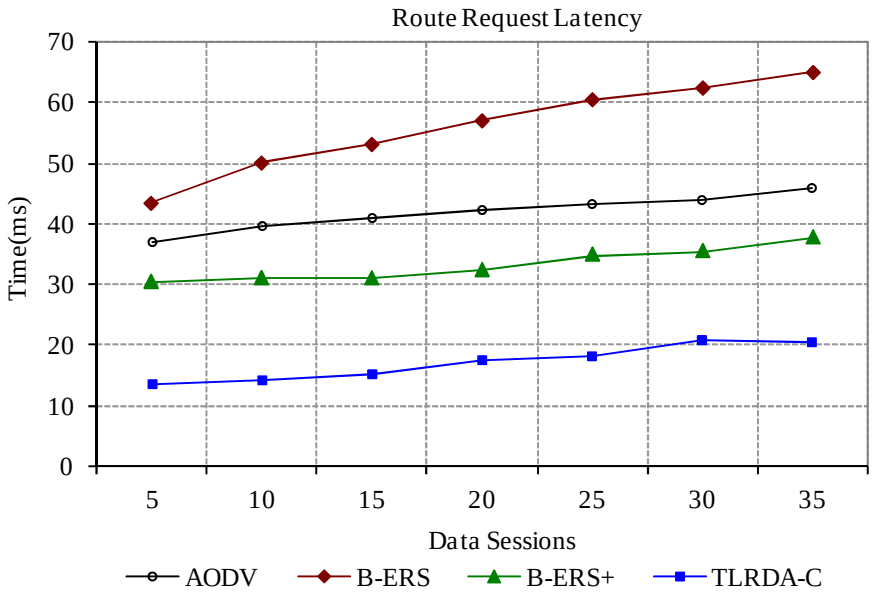

Figure A-9: Route request latency versus traffic load in a network of 70 nodes and 15m/s as maximum speed.

Overhead:

Figure A-10 depicts the routing overhead for all four algorithms. B-ERS+ incurs lower routing overhead than B-ERS due to the higher success rate of the catching process. B-ERS+ improvement increases with traffic load reaching 62% and 82% in heavy load over B-ERS and AODV respectively. TLRDA-C and B-ERS+ have almost the same routing overhead.

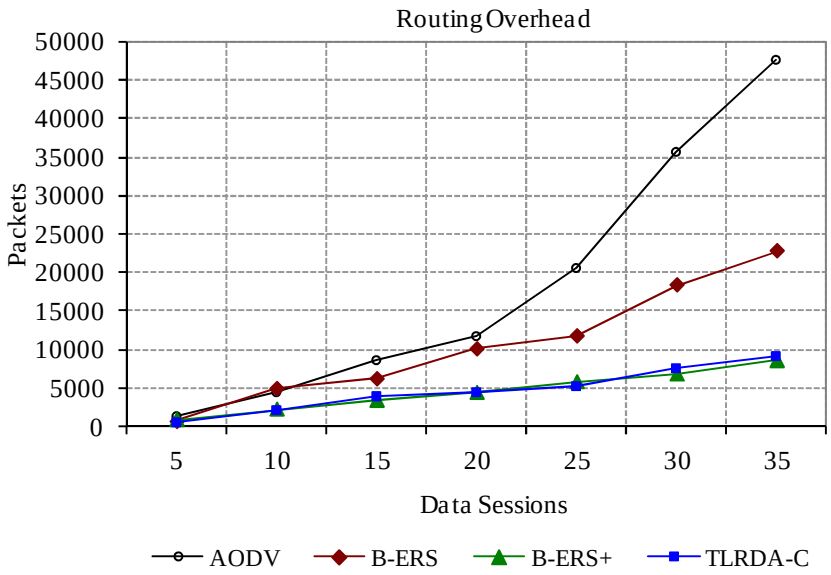

Figure A-10: Routing overhead versus traffic load in a network of 70 nodes and 15m/s as maximum speed.

Congestion:

B-ERS+ incurs less packet loss in the whole network compared to B-ERS as shown below in Figure A-11 because the network in B-ERS+ is less congested. The packet loss is increased with the increment of traffic load for all four algorithms. B-ERS+ improves packet loss by 29% to 71% over B-ERS, by up to 20% over TLRDA-C, and by 38% to 70% over AODV.

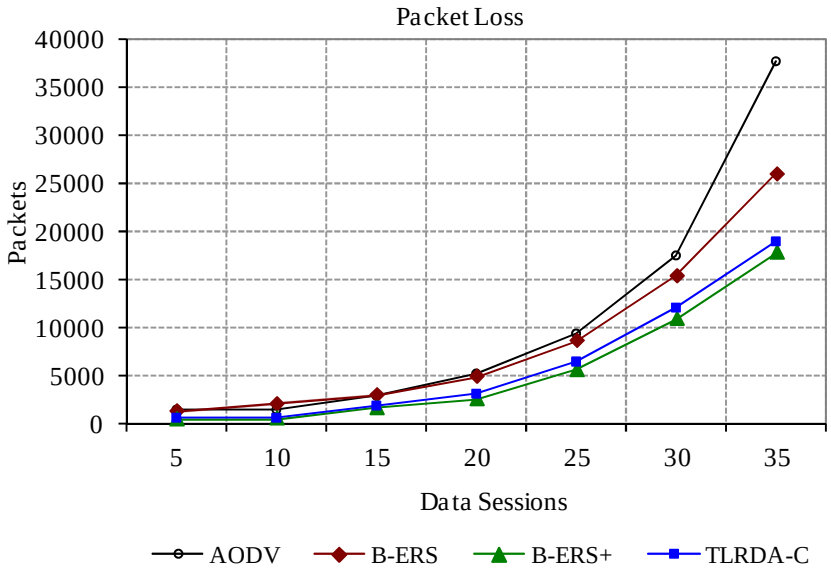

Figure A-11: Packet Loss versus traffic load in a network of 70 nodes and 15m/s as maximum speed.

A.3.3 Effect of mobility

Figures A-12 to A-16 were extracted from simulating the four algorithms for 900 seconds using networks of size 70 nodes in an area of 1000m x 1000m using six maximum speeds. The maximum speed takes one of the following values: 2, 5, 7, 10, 13, and 15m/s. The traffic load was fixed at 10 communication sessions.

Figure A-12 demonstrates network coverage as an indicator of the success rate of the catching process like the previous analyses. B-ERS+ improves the success rate of B-ERS regardless of speed by 80% to 86%. The success rates of TLRDA-C and B-ERS+ are very close to each other with a difference ranges from -9% to 15%.

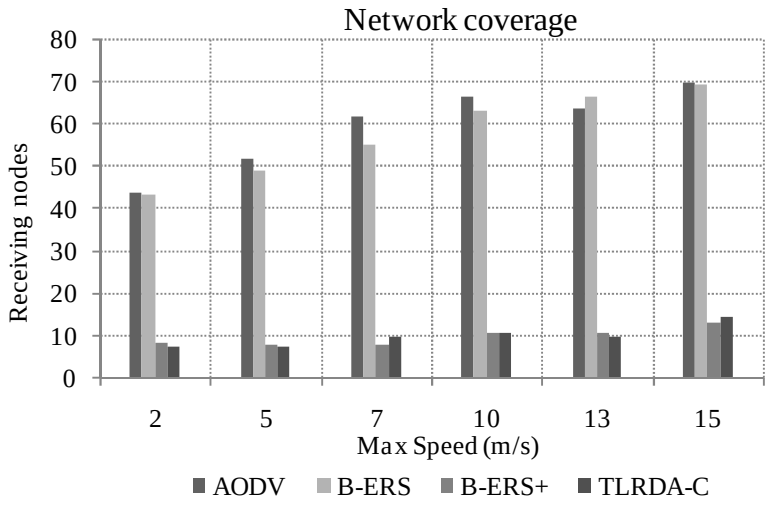

Figure A-12: Network coverage versus maximum speed in networks of 70 nodes and 10 communication sessions.

Latency:

TLRDA-C improves end-to-end delay over B-ERS+, B-ERS, and AODV as shown in Figure A-13. This improvement is due to the quick broadcasting within the neighbourhood region. TLRDA-C's improvement is from 23% to 40% over B-ERS+ while B-ERS+ improves end-to-end delay by 41% to 52% over B-ERS and by 31% to 42% over AODV.

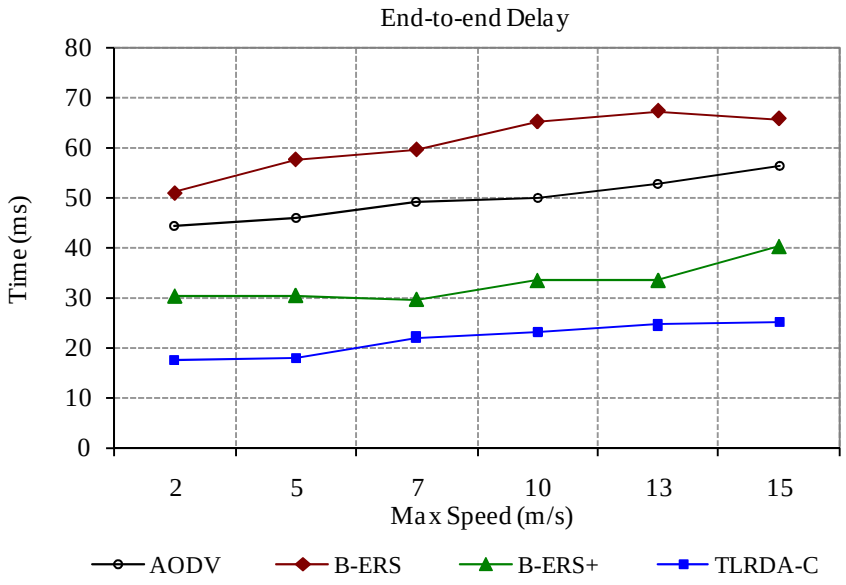

Figure A-13: End-to-end delay versus maximum speed in networks of 70 nodes and 10 communication sessions.

Route requests latency for TLRDA-C is lower than B-ERS+, B-ERS, and AODV regardless of speed as shown in Figure A-14 which improves network performance. As mentioned previously, this improvement is due to the higher success rate of TLRDA-C in the catching process and the quick broadcasting within the neighbourhood region. TLRDA-C improves the average of route request latency by 54% to 69% over B-ERS+. B-ERS+ improves route request latency by 26% to 42% over B-ERS because when the route request propagate further in the network the hop count increases which increase the amount of delay imposed. Moreover, B-ERS+ improves route request latency by up to 22% over AODV.

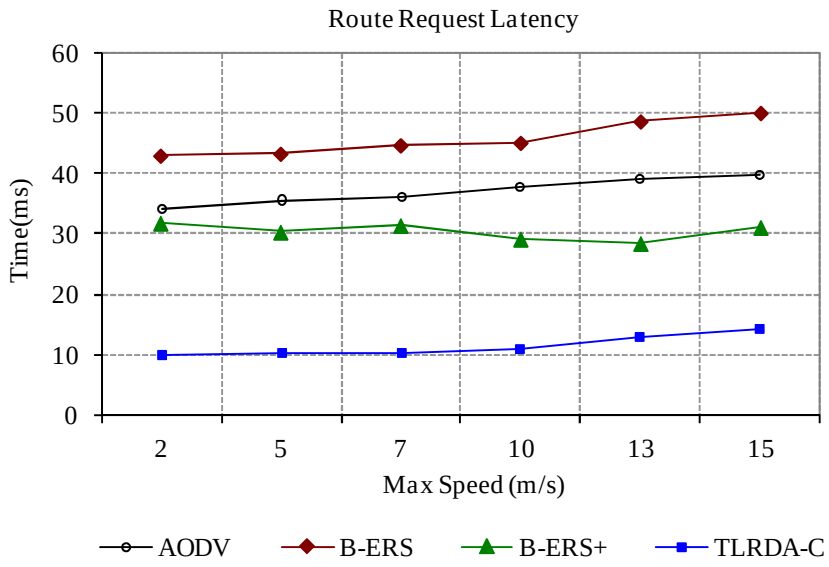

Figure A-14: Route request latency versus maximum speed in networks of 70 nodes and 10 communication sessions.

Overhead:

TLRDA-C and B-ERS+ incur low routing overhead; lower than B-ERS and AODV as shown in Figure A-15. Routing overhead increases more with fast networks regardless of the algorithm used. The improvement of the routing overhead in B-ERS+ ranges from 56% to 72% over B-ERS and by 44% to 60% over AODV. TLRDA-C and B-ERS+ routing overheads are relatively close.

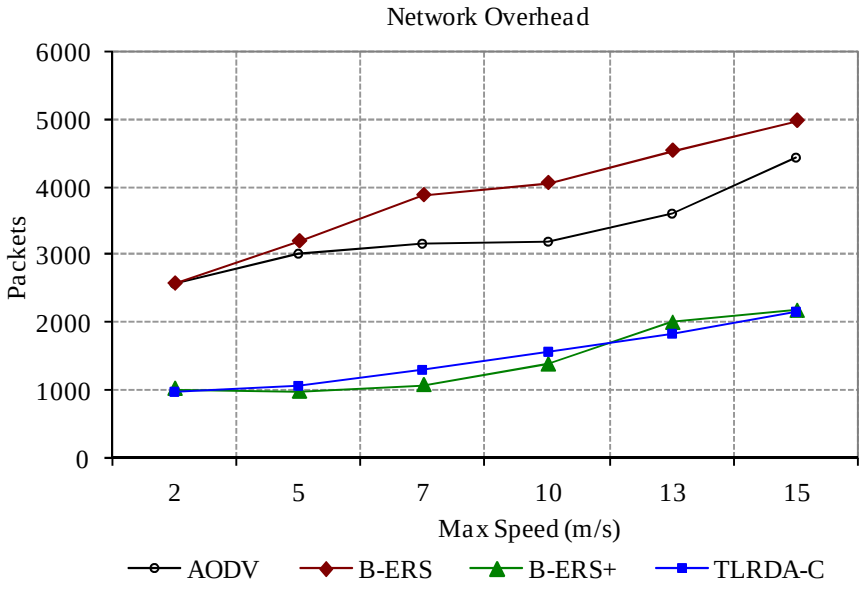

Figure A-15: Routing overhead versus maximum speed in networks of 70 nodes and 10 communication sessions.

Congestion:

TLRDA-C and B-ERS+ lose fewer packets compared to AODV and B-ERS as shown below in Figure A-16 because the networks are less congested in the case of TLRDA-C and B-ERS+. The packet loss is increased with the increment of maximum speed for all four algorithms. B-ERS+ improves packet loss by 68% to 76% over B-ERS and is by 65% to 86% compared to AODV while the difference between B-ERS+ and TLRDA-C ranges from -20% to 10%.

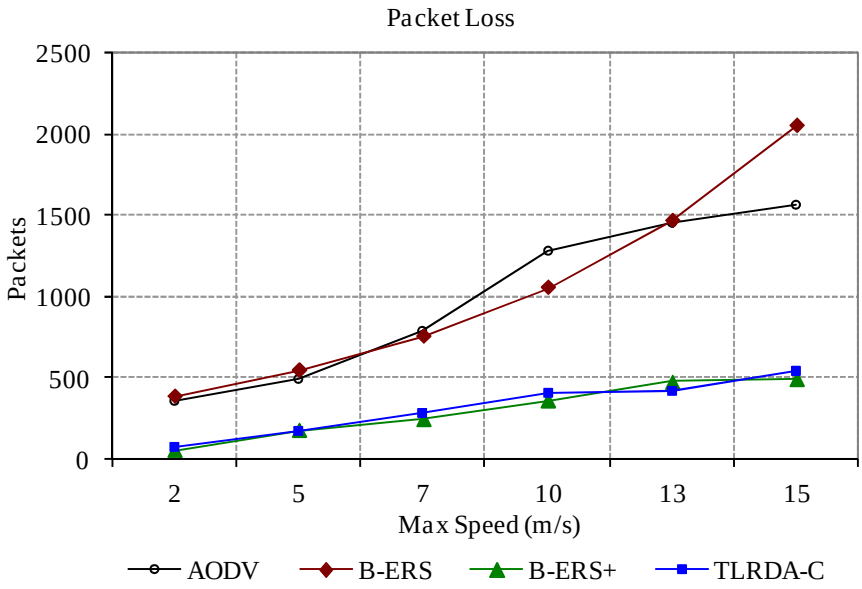

Figure A-16: Packet loss versus maximum speed in networks of 70 nodes and 10 communication sessions.

Looking at the three-performance analyses, our algorithm, TLRDA-C, outperforms AODV, B-ERS+, and B-ERS regardless of network size, traffic load, or speed in terms of end-to-end delay. B-ERS+ outperform B-ERS in all metrics used regardless of network size, traffic load, or maximum speed.

A.4 Summary of simulation results

Simulation experiments and analyses were conducted to study the performance of B-ERS+ while concentrating on the three-performance cases: network size, traffic load, or mobility. Almost the same behaviours were demonstrated by those simulation experiments and analyses for the following metrics: network coverage, end-to-end delay, route request latency, and routing overhead as well as packet loss depicted in Figure A-2 until Figure A-16.

B-ERS+ outperforms B-ERS algorithm by reducing end-to-end delay due to the reduction in network congestion. It also improves route request latency, routing overhead, packet loss due to

the higher rate of success in the catching process compared to B-ERS. Moreover, TLRDA-C outperform B-ERS+ in terms of route request latency and end-to-end delay while incurring almost the same overhead.

A.5 Conclusions

B-ERS achieves low success rate due to the early discard of chase packets which hinder the chasing process in the presence of mobility. B-ERS+ is a modification of B-ERS where the chase packets are allowed to travel in the network until the caching is insured. The simulation analyses show that B-ERS+ outperforms B-ERS by reducing the latency in terms of end-to-end delay and route request latency due to the success in freeing the network from unneeded route requests which reduces network congestion. B-ERS+ incurs lower overhead compared to B-ERS by reducing routing overhead and packet loss due to the higher rate of success in the catching process. TLRDA-C outperform B-ERS+ in terms of route request latency and end-to-end delay while incurring almost the same overhead.