MOBILITY-ADAPTIVE CLUSTERING AND NETWORK-LAYER MULTICASTING IN MOBILE AD HOC NETWORKS

ER INN INN

NATIONAL UNIVERSITY OF SINGAPORE

2006

MOBILITY-ADAPTIVE CLUSTERING AND NETWORK-LAYER MULTICASTING IN MOBILE AD HOC NETWORKS

ER INN INN
(B. Sc. (Hons.), UTM)

A THESIS SUBMITTED
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY
DEPARTMENT OF COMPUTER SCIENCE
NATIONAL UNIVERSITY OF SINGAPORE

2006

ACKNOWLEDGEMENTS

First, I would like to thank my thesis supervisor, Dr. Winston Seah Khoon Guan for his constant dedication, guidance, advice and inspiration. I greatly appreciate all the insights and criticism he have given me to improve the quality of this research and thesis. This thesis would not have been possible without his help and motivation.

Special thanks go to the National University of Singapore for sponsoring my PhD research. I would like to thank the Institute for Infocomm Research (I^2R) for providing a comfortable environment, sufficient resources and conference funding to me.

I would like to thank my fellow lab-mates of the past few years that accompanied me on this challenging path. Thank you for giving me valuable advice and help whenever I need them. They include Junxia, Kevin, Ricky, Choong Hock, Lixia, Qunying, Xiejing and Huixian. I would like to thank my dearest friends: Peiwen, Suchin, Keeyit, Benchin, Catherine, Sean doggie, Jimmy, Keone, Koktong, Yeechian, Ahming, Engela, and Josephine for being part of my life and also part of my PhD journey. Without your support and encouragements, I would not have gone this far.

Last but not least, I thank my family: Dad, Mum, Chinchin, Chinming, Sinyee, Andy, and Chenhui for unconditionally supporting my decision to study PhD, for selflessly loving me, for tirelessly encouraging me whenever I am feeling the sky is grey. For all the sacrifices they have done for me, I dedicate this thesis to my family.

TABLE OF CONTENTS

SUMMARY

Clustering has been used to provide a logical hierarchy for various network control functions like routing, location management, data replication, and so on. Forming and maintaining stable cluster structures in MANETs in view of the dynamic topology and scarce resources is very challenging. In this thesis, a mobility-based multi-hop clustering algorithm, namely Mobility-based D-Hop (MobDHop) clustering, is proposed to provide a long-lived and efficient cluster structure. MobDHop forms stable multi-hop clusters by introducing two mobility-related metrics, i.e. Local Variability and Group Variability as criteria to elect clusterheads and to maintain the cluster structure. MobDHop is able to capture and adapt to the existing mobility patterns in MANETs. Unlike other multihop clustering algorithms, the diameter of MobDHop is not fixed to a certain user-predefined parameter. Instead, the diameter of clusters formed by MobDHop is flexible and adaptive to mobility patterns in the network, requiring only one-hop neighbourhood information.

MobDHop has been validated using simulations and compared against two other algorithms, Lowest-ID (L-ID) Clustering and Maximum Connectivity Clustering (MCC). The results have shown that these three algorithms are comparable in performance when the Random Waypoint mobility was assumed in relatively small network. When group mobility or larger network size were assumed, MobDHop significantly outperformed L-ID and MCC algorithms in terms of cluster efficiency and stability. The analysis of message and time complexity of MobDHop shows that the number of packet transmissions per node per time step for MobDHop to operate correctly in MANETs is O(1), which is the same asymptotic bound for one-hop

clustering. It is shown in this analysis that multi-hop clustering is feasible in networks with high mobility without incurring prohibitive overhead.

Multicasting, on the other hand, is an essential mechanism to efficiently support group-oriented applications in resource-limited MANETs. A number of multicast routing protocols have been specially designed for MANETs. Most of these protocols were designed with small networks in mind. In view of this, designing a multicast solution for large MANETs, which is efficient, robust against mobility, adaptive to network conditions and more scalable, is another objective in this thesis. A cluster-based, GRoup-AdaPtivE (GRAPE) multicast routing protocol is proposed to provide scalable, robust and efficient multicast routing solution. GRAPE introduces a new two-tier multicast paradigm, which includes a two-tier multicast group management scheme and a two-tier multicast routing protocol. GRAPE works on top of the stable cluster architecture formed by MobDHop for increased protocol scalability. GRAPE was validated using the QualNet simulator over a large variety of scenarios and its performance was compared against of the On Demand Multicast Routing Protocol (ODMRP). Results show that GRAPE delivered larger percentage of multicast packets to receivers than ODMRP, in most scenarios, which it has been able to accomplish by incurring much lower data overhead. The better delay performance of GRAPE over ODMRP also makes GRAPE a better alternative for delay-sensitive applications. Simulation results show that GRAPE scaled gracefully with respect to network density, mobility, traffic load and multicast-related parameters.

To further enhance the multicast capability of MANETs, the Bandwidth-Optimized and Delay-Sensitive (BODS) multicast path setup algorithm, is also proposed in this thesis to construct per-source multicast mesh which is more optimal in terms of bandwidth consumption while retaining good delay performance. The performance of BODS was evaluated by integrating BODS into ODMRP in QualNet simulator. Results show that the BODS-enhanced ODMRP achieved similar or better packet delivery ratio as the original ODMRP by yielding a reduction of around 30% in data overhead. The delay performance was also improved by BODS integration especially in networks of high traffic load.

In short, this thesis contributes two novel network protocols for MANETs: (1) a clustering algorithm in search of MWIS which provides a stable and long-lived cluster structure to support various network functions such as unicast routing, multicast routing, security, resource management, and MAC optimization, and (2) a cluster-based multicast routing protocol which is more efficient, more robust and more scalable.

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ABAM	Associativity-Based Ad-hoc Multicast
ABR	Associativity-Based Routing
ADMR	Adaptive Demand-driven Multicast Routing
ALM	Aggregated Local Mobility
AMRIS	Ad-hoc Multicast Routing Protocol utilizing Increasing id-numberS
AMRoute	Ad hoc Multicast Routing Protocol
AODV	Ad-hoc On-demand Distance Vector
ARC	Adaptive Routing using Clusters
BODS	Bandwidth-Optimized and Delay-Sensitive Multicast Path Setup
CAMP	Core-Assisted Multicast Protocol
CBR	Constant Bit Rate
CBT	Core Based Tree
CDS	Connected Dominating Set
CGSR	Clusterhead Gateway Switch Routing
CoV	Coefficient of Variation
CRT	Cluster Residence Time
DCMP	Dynamic Core-based Multicast Protocol
DDM	Differential Destination Multicast
DMAC	Distributed Mobility-Adaptive Clustering
DSR	Dynamic Source Routing
DVMRP	Distance Vector Multicast Routing Protocol
ETC	Effective Topology Change
FGMP-RA	Forwarding Group Multicast Protocol-Receiver Advertising
FGMP-SA	Forwarding Group Multicast Protocol-Sender Advertising
GCA	Generalized Clustering Algorithm

GRAPE	GRoup-AdaPtivE Multicast Routing Protocol
GVar	Group Variability
HCNP	Hop Count to Nearest Participant
HDDM	Hierarchical Differential Destination Multicast
IGMP	Internet Group Management Protocol
IP	Internet Protocol
LAM	Lightweight Adaptive Multicast
LBM	Location-Based Multicast
LCA	Linked Cluster Architecture
LCC	Least Clusterhead Change
L-ID	Lowest-ID
MACT	Multicast ACTivation
MANET	Mobile Ad Hoc Network
MAODV	Multicast Ad-hoc On-demand Distance Vector
MCC	Maximum Connectivity Clustering
MCDS	Minimum Connected Dominating Set
MCEDAR	Multicast Core-Extraction Distributed Ad-hoc Routing
MDSR	Multicast Dynamic Source Routing
MIS	Maximum Independent Set
M-LANMAR	Multicast-enabled Landmark Ad Hoc Routing
MobDHop	Mobility-based *d*-hop Clustering Algorithm
MOSPF	Multicast Open Shortest Path First
MREP	Multicast Route Reply
MREQ	Multicast Route Request
MST	Minimum Spanning Tree
MWIS	Maximum Weight Independent Set
MZR	Multicast Zone Routing
NP	Non-Polynomial

NSMP	Neighbour Supporting Multicast Protocol
ODMRP	On-Demand Multicast Routing Protocol
PDR	Packet Delivery Ratio
PIM-DM	Protocol Independent Multicast- Dense Mode
PIM-SM	Protocol Independent Multicast- Sparse Mode
QoS	Quality of Service
ReF	Redundancy Factor
RPGM	Reference Point Group Mobility
RREP	Route REPly
RREQ	Route REQuest
RW	Random Waypoint
SMMRP	Scalable Multi-source Multicast Routing Protocol
SP	Shortest Path
UDG	Unit Disk Graph
WCA	Weighted Clustering Algorithm
WCDS	Weakly Connected Dominating Set
ZRP	Zone Routing Protocol

CHAPTER 1

INTRODUCTION

1.1 Introduction

The recent rise of mobile devices has aroused unprecedented research interest in mobile wireless networks. Conventional wireless networks are operating on some fixed backbone network with radio base stations, where only the last hop to the users is wireless. As wireless networks proliferate, a new variant of mobile wireless network that does not rely on any fixed infrastructure and can be setup in an ad hoc manner emerges. This variant is widely known as mobile ad hoc network (MANET) [1]. MANETs are collections of autonomous and mobile network devices (nodes) interconnected by multihop wireless communication paths without any centralized control. MANET nodes have to act as routers to discover and maintain routes to other MANET nodes. This is uncommon in traditional computer networks as routers are usually specialized devices that determine the best path for forwarding data packets. Since there is no special requirement except a set of independent mobile stations in order to deploy a MANET, these networks can be deployed and re-deployed spontaneously at anytime and anywhere. They are usually self-creating, self-organizing, and self-administering [2].

Due to the fact that MANET nodes can move freely, the MANET topology may change rapidly and unpredictably. Besides, adjustment of transmission and reception parameters such as power may also impact the topology. The dynamic topology induces challenges to routing protocol design which has been based on static topology in conventional wired networks. Apart from dynamic topologies, wireless links that connect MANET nodes are usually bandwidth-constrained and their capacity may vary over time. Most if not all MANET nodes are relying on a limited energy source for power. Therefore, power consumption becomes another critical issue in the protocol design of MANETs. Security issue has been a great concern in MANET research since physical security is limited due to the wireless medium used in data transmission. However, MANETs are still desirable since it can meet the demand of certain applications like military applications that requires immediate deployment and survivability.

The US Department of Defence, in particular DARPA, pioneered the research in MANETs with the deployment of Packet Radio Network (PRnet) in 1972 [3]. The motivation of PRnet is to relieve the network from relying on base stations due to the fact that the deployment of base stations is difficult and almost impossible in hostile environments. Furthermore, the network is subject to failure if one or several base stations are destroyed. The mobility of nodes is also limited as the mobile nodes must be in the transmission range of base stations. On the other hand, MANET, with its distributed network architecture and broadcast radio, is more suitable for the military deployments. To overcome the limited radio transmission ranges, nodes are equipped with the ability to act like a router and to forward information on behalf of others, i.e. multi-hop communications as shown in Figure 1.1 unlike the last-hop wireless networks as shown in Figure 1.2. Driven by the need to establish multihop communications in an ad hoc manner, a large number of unicast routing protocols has been proposed for MANETs. A detailed review of unicast routing protocols for MANET can be found in [4].

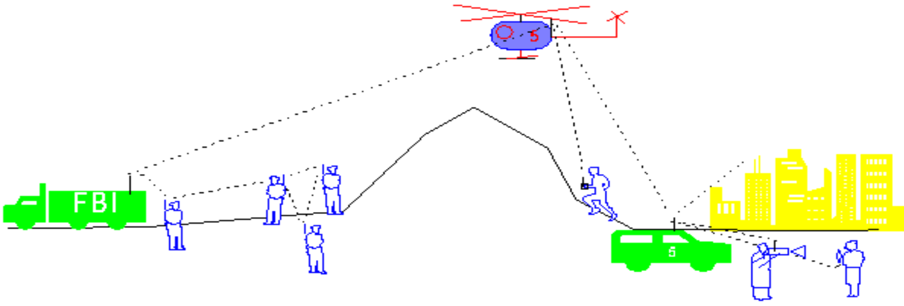


Figure 1.1 Multihop mobile wireless networks, a.k.a. MANETs.

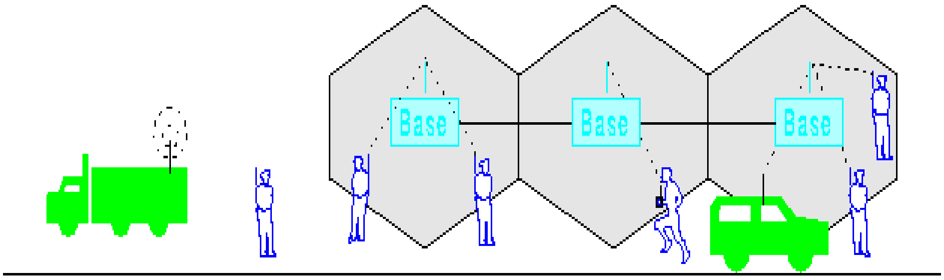


Figure 1.2 Single-hop mobile wireless networks, a.k.a. standard cellular networks.

Subsequent DARPA projects like SURAN in 1983 [5], Global Mobile (GloMo) Information Systems program in 1994 [6], and the on-going Land Warrior program [7] and its deployment [8] involve a larger number of mobile devices and a wider region. Apart from military applications, large-scale commercial applications of MANETs also start to blossom with the proliferation of wireless technology. Businesses start to envision large-scale commercial applications like smart vehicular system [9] and a radio dispatch system for public transportation system [10]. As the scale of MANETs continues to grow, one of the most critical design elements of MANET protocols is their applicability in large-scale deployments, i.e. the protocol scalability [11][12][13]. Forming a logical hierarchical network organization by clustering is one of the common approaches to increase protocols' scalability [13]. With group-oriented communications likely to dominate in large-scale MANETs applications, mobile hosts will also exhibit coordinated moving patterns such as group mobility. For example, police officers are divided into teams to conduct coordinated search operation for criminals in hiding, or rescue teams searching for victims in disaster-stricken areas. This motivates the need to exploit group mobility pattern in clustering so that a stable logical hierarchical network organization can be formed and maintained to increase protocol scalability.

Group-oriented and collaborative applications [14] like content-based resource-discovery, multi-party video conferencing, multi-player networked online gaming, corporate communications, distance education, and distribution of software, stock quotes broadcast and news broadcast are likely to become killer applications in MANETs. This suggests that the traffic in MANETs could consist of those that are destined for a group of nodes. In view of this, multicast [15] will be useful in MANET. A single stream of data can be disseminated to multiple recipients without clogging the networks by using the multicast mechanism as each packet is transmitted only once by the source and duplicated whenever necessary. A number of multicast routing protocols have been proposed for MANETs and most of these protocols assume that the network topology is flat. However, the deployment of large-scale MANET for military and commercial applications may consist of hundreds or possibly thousands of nodes. This raises the scalability issue of multicast routing protocol that requires further investigation. In this chapter, a brief overview on clustering issues in MANETs will be presented in section 1.2. Issues on network-layer multicasting in MANET will be examined in section 1.3. This section includes a discussion on the current status of research development and related research issues. This will be followed by the objectives, scopes and contributions of this thesis in section 1.4 and 1.5 respectively.

1.2 Clustering Issues in MANETs

Clustering algorithms are widely used in communication networks to organize nodes into logical groups (clusters) in order to provide a hierarchical network organization. A subset of nodes are selected from each cluster as representative nodes to serve as the network backbone for providing essential network control function such as address assignment, routing, network management, security and others. In multicast routing, the routing and group membership tables could grow to an immense size if all nodes store complete multicast routing details for a large MANET. This raises scalability issues in the flat topology assumed by previous MANET multicast routing protocols. Apart from protocol scalability, clustering may be used to facilitate the implementation of spatial reuse, location management, network management, security provision and QoS support. Spatial reuse can be implemented by managing wireless transmission among member nodes to reduce channel contention.

There have been a number of clustering algorithms proposed to build the logical hierarchical organization in MANETs. There are mainly two different approaches to perform clustering: (1) Minimum Connected Dominating Set (MCDS) construction and (2) Maximum Weight Independent Set (MWIS) construction. Some of the eminent clustering algorithms from both approaches will be reviewed in Chapter 2.

Forming a stable cluster structure in a mobile environment remains as a challenging agenda in the design of MANET clustering algorithms. Apart from the instability of cluster structure, most previously proposed clustering algorithms only form one-hop clusters in MANETs where the maximum diameter of the cluster equals two. Therefore, they are more suitable for relatively smaller and denser MANETs in which most of the nodes are within direct transmission range of clusterheads. However, these algorithms may form a large number of clusters in relatively large MANETs and eventually lead to the same problem as in a flat architecture. A very few multihop clustering algorithms were proposed in the literature. These approaches form cluster structure which is less stable as the algorithms do not take mobility into consideration during the formation and the maintenance of their multihop cluster structure. Moreover, these algorithms involve flooding of the clustering information up to multiple hops. The flooding coverage is usually defined by the maximum value of the radius of clusters formed. This incurs high signalling overhead which is extremely prohibitive in MANETs. The diameter of the clusters formed by these algorithms is also fixed and subject to a user-defined parameter.

1.3 Multicast Routing Issues in MANETs

Imagine a scenario where a commander intends to send critical battlefield strategy to a few squads of soldiers on the field via MANET. If unicast technique is deployed, the commander's device will repeatedly send out duplicate sets of data to all recipients. This will not only waste the scarce bandwidth in the MANET, but also cause network congestion and possibly a significant delay in data transmission. Moreover, the duplicate copies of data may congest the network and bring it down. To overcome this, multicast technique is introduced in the late 80's by Steve Deering [15]. Multicasting is the transmission of datagram (packets) to a group of hosts identified by a single destination address. A multicast packet is typically delivered to all members of its destination host group with the same reliability as regular unicast packets.

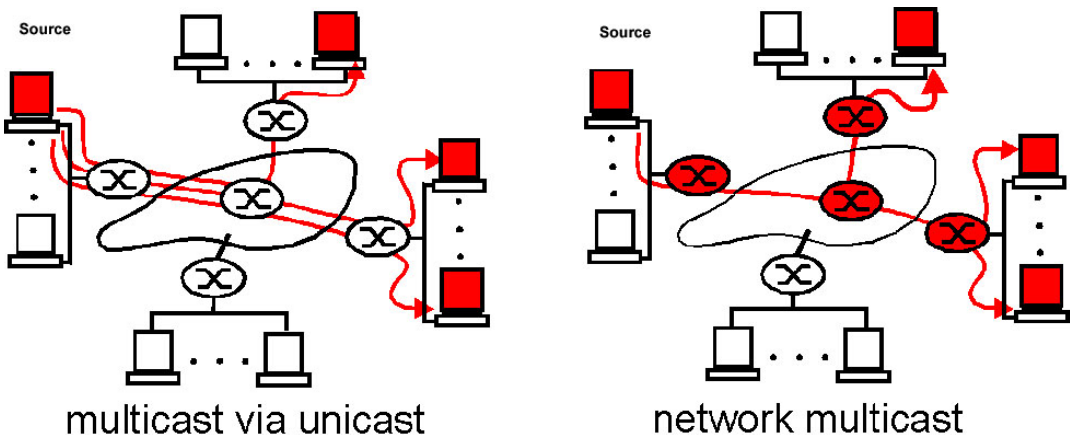


Figure 1.3 Unicasting vs. multicasting.

Multicasting is intended for group-oriented computing and its use within a network has many benefits. It is more efficient as it builds a multicast delivery infrastructure, which allows the multicast source to transmit only one copy of the information and the intermediate nodes will duplicate the information when needed. Only nodes that are part of the targeted group will receive the information. Figure 1.3 shows the difference between unicasting and multicasting. These features are particularly important in MANETs which have limited resources such as bandwidth and battery power.

Setting up a multicast delivery infrastructure is an essential component in network-layer multicasting [16]. There are several approaches being adopted to construct a multicast delivery infrastructure. The most straightforward way is to build a routing tree by adding one participant at a time, using the shortest path algorithm [17]. New participants are connected along a shortest path to the source in the existing tree. While the shortest path tree between the source and receivers guarantees that multicast packets will be delivered as fast as possible, it does not necessarily result in a tree that optimizes the network resources such as bandwidth. This approach builds per-source tree. Thus, it is more suitable for one-to-many communication. The second approach is to construct a shared tree to distribute the traffic from all senders in the group, regardless of the senders' location, and to minimize the total weight of the tree. Hence it optimizes the use of network resources. The problem of finding such a minimum-weighted tree that spans all multicast users is usually modeled as the Steiner Tree problem in the networks [18]. Due to the complexity in finding Steiner tree, Minimum Spanning Tree (MST) [17] algorithm is commonly used to provide an approximation. The path length between sources and destinations may not be the shortest in the network.

The multicast routing protocol has two main responsibilities: (1) to collect and maintain state information that can be used by the multicast routing algorithms for path selection, (2) to select the most appropriate path among the various paths available using a path selection algorithm [19]. As a result, a number of well-defined multicast routing protocols such as Distance Vector Multicast Routing Protocol (DVMRP) [20], Multicast Open Shortest Path First (MOSPF) [21], Core-Based Tree (CBT) [22], Protocol Independent Multicast –Dense Mode (PIM-DM) [23] and Protocol Independent Multicast –Sparse Mode (PIM-SM) [24] were introduced and deployed in Internet Protocol (IP) networks. Multicasting in this context is known as IP multicasting. However, IP multicast routing protocols are not well-suited for MANETs. The multicast problem is more complicated due to the frequent topology changes in MANETs. The difficulty in implementing IP multicasting in wireless networks has been discussed in [25]. Existing IP multicast routing protocols have been designed for fairly static networks and are based on two basic principles [26]:

i) Creation of delivery trees that control the path that IP multicast takes to deliver traffic to all receivers.

ii) Use of preexisting routing infrastructure such as link state or distance vector techniques for the maintenance of such trees.

However, the validity of these principles is undermined by the dynamic nature of MANET topologies. As pointed out in [27], frequent topology changes in MANETs resulting from their unconstrained mobility characteristics trigger the reconfiguration of multicast delivery trees assumed in IP multicasting. This results in excessive channel and processing overhead as well as frequent loss of data packets.

Apart from multicast efficiency, the design of multicast routing protocols for MANETs must also satisfy another key demand, which is the robustness against mobility [28]. In other words, multicast routing protocols for MANETs have to be efficient by incurring low data and control overhead, as well as robust by being resistant against topology changes. The widespread of mobile devices and the envisioned large-scale MANETs prompt the need to investigate into multicast protocol scalability issue. There is on-going effort in IRTF MANET WG [29] in order to establish a standard framework for defining, evaluating and comparing protocol scalability in MANETs. The scalability of a protocol in MANETs is a measure of its ability to maintain good performance, which is defined by certain performance metrics, as some parameters of the network increase to very large values. It is possible to have more than one metrics of interest in the determination of protocol scalability with respect to a given parameter in a particular environment. The authors suggested three methods to evaluate the protocol scalability but they are yet to arrive at a conclusion where fair comparison can be achieved.

Another important consideration for MANET multicasting is quality-of-service (QoS) support. In critical missions such as military or emergency operations, multicast mechanisms, though attractive in saving network resources, may not be well-suited if successful or in-time packet delivery cannot be guaranteed. In other applications, such as video/audio conferencing, excessive loss of packets or unpredictable end-to-end delay may distort the original information. Therefore, QoS routing that can provide routes which satisfy QoS requirements of specific multicast applications is desirable, e.g. [30].

It is challenging to design a single cure-all multicast routing protocol for MANETs. Prior works in MANETs show varied performance under different environments. Existing multicast routing protocols for MANETs are designed based on different assumptions and each is only suitable for specific network conditions.

1.4 Objectives and Scopes of the Research

The objective of this research was twofold: (i) to design a new, fully distributed clustering algorithm that adaptively takes mobility pattern into consideration in order to construct a stable and long-lived cluster structure in MANETs, and (ii) to design a new multicast routing protocol which works on top of a pre-existing cluster structure for MANETs.

Since the cluster structure will act as an underlying logical hierarchical control structure to increase multicast protocol scalability for MANETs, the new clustering algorithm must form cluster with high stability. The design of this clustering algorithm must be distributed, fully localized where only localized information is required to perform clustering and must not involve network-wide flooding. It should incur as minimum clustering overhead as possible in view of the scarce resources in MANETs. Optimal clustering may not be achieved, but the algorithm should be able to form valid cluster structure if any exist that is as stable as possible.

The multicast routing protocol must be loop-free and independent of any unicast routing protocol. This cluster-based multicast routing protocol should satisfy important protocol requirements such as multicast efficiency, protocol robustness against mobility and protocol scalability. Multicast efficiency is defined as the gain of multicast in terms of network resource consumption compared to unicast [31]. In other words, this protocol should deliver as many data packets as possible to the set of receivers by incurring as little redundant data transmissions as possible.

Protocol robustness is defined in this research as the ability of the protocol to maintain the satisfactory performance in the presence of mobility. This indicates that the protocol should be able to minimize packet loss due to mobility. Protocol scalability is defined as the ability of the protocol to support the continuous increase of the network parameters (such as network size, network density, mobility rate, data generation rate) without degrading network performance [32]. This kind of absolute protocol scalability [32] is very hard to be defined in mobile environments. Therefore, the "Weak Scalability" notion as suggested in [29] was adopted in this research. "Weak Scalability" refers to the comparison of the performance metrics of interest with respect to a given range of the network parameter of interest in a particular environment. In literature, the performance metrics of interest in a MANET multicast routing protocol include the packet delivery ratio, the delay performance, and the routing overhead. Meanwhile, the network parameters of interest include the network density, network size, mobility rate, data generation rate and multicast-related parameters. There is also a large group of works done in designing energy-efficient multicast by using power control method in MANETs [33]. However, energy-efficiency issue was not considered in this research due to the extra requirements on mobile devices such as power control capability and the additional complexity of the power control mechanism in the presence of network mobility. Apart from energy-efficiency, QoS issues were also not considered in this research.

It is important to validate and evaluate the performance of the proposed protocols. The use of network simulation is a widely-accepted practice in the wireless networking field for protocol evaluation. Therefore, this research also consisted of the implementations of the proposed protocols in widely used network simulators, such as NS-2 [34] and QualNet [35]. The performance of the proposed schemes should be evaluated by simulating various network scenarios that can represent various real-life situations. The performance of both the clustering algorithm and multicast routing protocol should be compared against existing approaches reported in literature.

1.5 Contributions of the Research

This research may lead to the birth of blueprints of two useful network protocols for MANETs: (1) a clustering algorithm in search of MWIS which provides a stable and long-lived cluster structure to support various network functions such as unicast routing, multicast routing, security, resource management, and MAC optimization, and (2) a cluster-based multicast routing protocol which is more efficient, more robust and more scalable. These blueprints may be further enhanced and practically implemented in future networking devices to support real-world deployment of large MANETs.

In this research, a mobility-based *d*-hop (MobDHop) clustering algorithm [36] was proposed to form a two-tier, multihop cluster structure for MANETs in order to support multicast routing function with increased protocol scalability (Chapter 3). MobDHop is a mobility-adaptive multihop clustering algorithm that forms and maintains clusters with flexible diameter. The diameter of the clusters formed by MobDHop is flexible and adaptive to the node mobility pattern in MANETs. However, users can define parameter *d*, in order to control the diameter of clusters from growing too large. MobDHop is fully distributed where it only requires one-hop neighborhood information for its correct operation. In this research, MobDHop was evaluated via network simulations to verify the high quality of cluster structure formed, i.e. stable and mobility-adaptive (Chapter 4). Its performance was compared against another two well-known clustering algorithms, namely, Lowest-ID and Maximum Connectivity Clustering. It had been shown by simulations that MobDHop is a more suitable clustering algorithm in MANET due to its adaptation to mobility. Another contribution of this research is an analytical investigation on multihop clustering overhead and time complexity. It had been shown in this research that the overhead incurred by multihop clustering has a similar asymptotic bound as one-hop clustering while being able to reap the benefits of multihop clusters [37][38]. It was also shown in this research that the cluster structure formed by MobDHop algorithm can support unicast routing function. A new variant of AODV protocol, namely MobDHop-AODV was proposed in this research (Chapter 4). MobDHop-AODV works on top of the cluster structure formed by MobDHop and utilizes the cluster membership knowledge of clusterheads to avoid unnecessary network-wide flooding in MANETs.

This research also proposed a new cluster-based multicast routing protocol, namely Group-AdaPtivE (GRAPE) protocol that works on top of a pre-existing stable logical cluster structure (Chapter 5). In GRAPE, a new multicast group management scheme that spreads the load of group management among source and clusterheads was introduced. GRAPE also consists of a two-tier multicast forwarding infrastructure that lends more flexibility and scalability to multicast routing. The upper-tier multicast communication structure connects source to clusterheads that are interested to join the multicast communication on behalf of their members. The packets dissemination for upper-tier structure is done in a more efficient Steiner-like mesh which is constructed by a new multicast path setup algorithm, namely Bandwidth-Optimized and Delay-Sensitive (BODS) algorithm (Chapter 6). The BODS [39] multicast algorithm is a fully distributed multicast path setup algorithm that uses Nearest-Participant Heuristic. This algorithm aims to construct a forwarding structure which is more optimal in terms of multicast efficiency without compromising the delay performance. The lower-tier multicast communication structure connects clusterheads and its members that join the multicast group. Clusterheads dynamically select a suitable forwarding scheme, i.e. either cluster broadcasting or stateless multicasting to forward packets to its members based on the group membership characteristic within their clusters. It may switch from one scheme to another if the group membership within cluster changes.

The performance of both BODS and GRAPE were evaluated using the simulation approach. It had been shown in this research that BODS enhances the multicast delivery structure by providing better multicast efficiency without sacrificing protocol robustness and delay performance. To show that GRAPE satisfies the design properties such as multicast efficiency, robustness and protocol scalability, an extensive series of simulations with different network configurations were conducted. The performance metrics of interest were evaluated over a set of network parameters of concern. As the "Weak Scalability" notion was adopted in this research, the performance of GRAPE with respect to these network parameters was compared relatively to that of On-Demand Multicast Routing Protocol (ODMRP), a well-known multicast routing protocol in MANETs. The simulation results showed that GRAPE provided a better packet delivery ratio, while utilizing much lower data overhead and incurring much lower delivery latency in various network scenarios simulated. The simulation results not only verified the required properties of GRAPE but also formed the basis for further investigation of other multicast routing issues which are beyond the scope of this research, such as energy efficiency, Quality of Service (QoS) support and probabilistic reliability.

1.6 Thesis Organization

This thesis is subdivided into eight chapters. Chapter 2 presents the literature survey of clustering algorithms and network-layer multicasting in MANETs. A detailed survey on different clustering algorithms in MANETs is presented following a brief analysis of the desired properties of clustering algorithm that can provide a good logical hierarchical structure to support various network control function including multicast routing. Most of the existing multicast routing protocols in MANETs will be briefly discussed and analyzed in order to justify the need of a new multicast routing protocol. Chapter 3 proposes a new clustering algorithm which provides a logical two-tier hierarchy in MANETs. A mobility-adaptive clustering algorithm, namely MobDHop, is proposed to organize a MANET into a number of non-overlapping, variable-diameter clusters. Chapter 4 presents the results of both empirical and theoretical analysis of the performance of MobDHop. While the focus of this thesis is on clustering and multicast routing, a typical network will definitely contain unicast traffic and a clustering algorithm must be able to support both types of traffic. Hence, for completeness, we also provide a simple study on the use of MobDHop clustering algorithm to support unicast routing protocols in MANETs. Simulation results and discussions on the integration of MobDHop into a well-known unicast routing protocol, Ad hoc On-demand Distance Vector (AODV) protocol is presented in this chapter. Chapter 5 presents the design of a cluster-based multicast routing protocol, namely Group-AdaPtivE

(GRAPE) multicast routing protocol, which works on top of a clustered MANET to achieve the desired properties of multicast efficiency, protocol robustness, and scalability. Chapter 6 presents a new algorithm, Bandwidth-Optimized and Delay-Sensitive (BODS) multicast path setup algorithm, which builds a Steiner-like multicast forwarding structure for efficient multicast delivery. Simulation results and discussions on the integration of BODS into a well-established multicast routing protocol, On-Demand Multicast Routing Protocol (ODMRP) is also presented in Chapter 6. The BODS algorithm was also integrated into GRAPE multicast routing protocol in this research. Chapter 7 presents simulation results and discussions of BODS-integrated GRAPE and the performance of BODS-integrated GRAPE was compared against the performance of ODMRP. Finally Chapter 8 concludes this thesis and discusses future work.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

Clustering approach is one of the alternatives to increase protocol scalability in large-scale MANETs. A review on various clustering algorithms previously proposed in MANETs is presented in the following section. As discussed in Section 1.3, multicast is an extremely useful abstraction in view of the scarcity of network resources in MANETs. This research focused on the explicit multicast support at the network layer in MANETs. In this chapter, a detailed review will be presented on the network-layer multicast problem in MANETs. The network-layer multicast problem consists of (1) multicast group management, (2) multicast path setup algorithm, and (3) multicast routing protocols.

2.2 Clustering Algorithms for MANETs

Clustering algorithms are widely used in communication networks such as the Internet, ATM networks and cellular networks to organize nodes into logical groups (clusters) in order to provide an underlying hierarchical network organization. A subset of nodes are selected from each cluster as representative nodes to serve as the network backbone for providing essential network control function such as address assignment, routing, network management, security and others. Clustering is proposed to be used to facilitate the implementation of spatial reuse, location management, network management, security provision and QoS support. Spatial reuse can be implemented by managing wireless transmission among member nodes to reduce channel contention [40]. Clustering also provides controlled access to the channel bandwidth and scheduling of nodes in each cluster in order to provide QoS support [41] in MANETs. As to the network management aspect, the Ad hoc Network Management Protocol (ANMP) [42] adopts three-level hierarchical cluster architecture for efficient network data collection. Streenstrup [43] summarizes that cluster-based control structures can be used in MANETs to improve efficiency of resource use in the following manners:

i) Reduce channel contention by managing wireless transmissions among multiple nodes.

ii) Reduce network diameter by forming routing backbones.

iii) Reduce network state information in quantity and variability.

Some relatively large MANETs (e.g. hundreds or possibly thousands of nodes per autonomous system) may need to store complete routing details for an entire network topology. In multicast routing, the routing tables could grow to an immense size if all nodes store complete multicast routing details for a large MANET. This raises scalability issues in the flat topology assumed by most of the existing MANET multicast routing protocols. Clustering algorithms are proposed in MANETs as one of the approaches to address the scalability issue. In general, clustering can provide the following benefits for large networks in terms of routing [44]:

i) Scalability – If a flat structure is used in large networks, routing tables and location registers would grow to an immense size. Therefore, partitioning the network into multiple clusters can limit the size of routing tables.

ii) Reduced signaling traffic – Detailed topology information for a fraction of the network (cluster) is only exchanged among local cluster members whereas aggregated information is distributed between neighboring clusters in the higher hierarchical level.

There have been a number of clustering algorithms proposed to build the logical hierarchical structure in MANETs. There are mainly two different approaches in clustering: (1) MCDS construction and (2) MWIS construction. Some of the eminent clustering algorithms for both approaches will be reviewed in the following sections.

2.2.1 Properties of Clustering Algorithms

Clustering becomes more complicated when dealing with mobile ad hoc networks (MANETs) due to its dynamic topology. Since there is no central control in MANETs, clustering must be performed in a fully-distributed, real-time and mobility-adaptive fashion. Clustering algorithm in MANETs should be able to maintain its cluster structure as stable as possible while the topology changes [40]. This is to avoid prohibitive overhead incurred during clusterhead changes. There are some techniques suggested to reduce clusterhead changes, e.g. the Least Clusterhead Change [45] algorithm suggests that a clusterhead change will not occur until another clusterhead comes into the direct transmission range of the existing clusterhead. There are several important properties that must be taken into account when designing a clustering algorithm for MANETs, i.e. cluster architecture, cluster coverage, cluster initialization and cluster maintenance.

2.2.1.1 Cluster Architecture

Most clustering schemes for MANETs are based on the notion of clusterhead. The clusterhead may be dynamically selected from the set of nodes. Clusterhead acts as a local coordinator of transmissions within the cluster. Due to lack of special capabilities, clusterheads may become a bottleneck in the system since it needs to do extra work. The selection of clusterheads is very important. This is known as centralized cluster architecture since each cluster has a central controller, i.e. clusterhead. Examples of these clustering schemes include the Lowest-ID [46][47], the Maximum-Connectivity clustering (MCC) [48], Distributed Mobility-Adaptive Clustering (DMAC) [49], Max-Min *d*-clustering [50], Weakly Connected Dominating Set (WCDS) [51], MOBIC [52], Mobility-based clustering (MBC) [53], Least

Clusterhead Change (LCC) [45], Passive clustering [54], and Adaptive Routing using Clusters (ARC) [55]. There are different criteria in selecting the clusterheads, such as node identifier in the Lowest-ID algorithm, node degree in MCC, combined metric in DMAC, and Aggregate Local Mobility (ALM) in MOBIC. In contrast, some schemes eliminate the requirement for a clusterhead. Since there is no notion of clusterhead, each node within a cluster is treated equally. This avoids vulnerable centers and hot spots of packet traffic flow. However, these algorithms lack of centralized control which may be useful to support different network functions. Some example schemes are 1-clustering (cliques) [56], *k*-clustering [57], (α, *t*)-clustering [58] and adaptive-clustering [40].

2.2.1.2 Cluster Coverage

Most of the clustering algorithms proposed for MANETs implement one-hop clustering, e.g. [40], [46], [47], [48], [49], [52], and [55]. One-hop clustering requires each pair of nodes in the same cluster to be at most two hops apart from each other, i.e. each member is at most one-hop away from clusterhead. One-hop clustering has the following properties:

i. There is a clusterhead at the center of a cluster and the clusterhead can communicate with any node in the cluster within one hop.
ii. No clusterheads are directly linked.
iii. Any two nodes in a cluster are at most two hops away.

However, some algorithms form clusters that allow longer hop-path with respect to the clusterhead, e.g. *k*-clustering [57], (α, *t*)-clustering [58], Max-Min *d*-clustering [50], and Mobility-Based Clustering (MBC) [53]. The properties owned by one-hop clustering may not be valid for other multihop clustering algorithms. For example, clusterheads in the multihop clusters formed by Max-Min *d*-Cluster may not be the center of its cluster. Some clusterheads may be a leaf node or border node.

2.2.1.3 Cluster Initialization

The first phase of clustering is usually cluster initialization or cluster setup. This is accomplished by choosing some nodes that act as coordinators of the clustering process (clusterheads) or selecting certain nodes to form a backbone in facilitating data transmission across the network. Then a cluster is formed by associating those nodes with their neighbors. Therefore, the issues that need considerations in this phase include the selection of clusterheads, the boundary of individual cluster, the coverage of each cluster, the formation of the overlapping cluster or non-overlapping cluster, as well as the selection of gateway nodes. Some algorithms require the network topology to be static during the cluster initialization, e.g. [40], [46], [48], and [56].

2.2.1.4 Cluster Maintenance

After the clusters are formed, some techniques need to be adopted in maintaining the cluster organization. As the cluster members are mobile, it can move from one cluster to another. Therefore, managing cluster membership is the main challenge in maintaining hierarchical organization in MANETs. Cluster reorganization is an expensive operation which may involve re-election of clusterhead, hand-over of information to a new clusterhead, as well as re-associating the nodes to a new clusterhead. Therefore, the main design goal of clustering algorithm is to minimize cluster reorganizations. However, cluster reorganization is unavoidable in presence of mobility. Some clustering algorithms assume the reorganization to be done in periodical manner [50]. Most of the clustering algorithms proposed in the literature do not suggest any maintenance scheme.

2.2.2 Existing Clustering Algorithms for MANETs

There are mainly two approaches to form local hierarchy: (1) through the construction of MCDS, and (2) through the construction of MWIS. A number of clustering algorithms based on both MCDS and MWIS construction approaches will be reviewed in the following sections. MWIS algorithms mainly differ from one another in the criterion they use to elect clusterheads, e.g. node identifier (ID), node degree, etc. The Adaptive Clustering Algorithm [40] suggests the exclusion of clusterhead in clusters in order to avoid vulnerable centres and hot spots of packet traffic flow. However, clusterheads can act as the central controller to efficiently provide various network management functions, QoS support as well as routing function in MANETs. Therefore, flat cluster architecture is less suitable and was not adopted in this research.

2.2.2.1 Minimum-Connected Dominating Set Approach

The topology of MANET is usually modelled as a unit-disk graph (UDG) [59], a geometric graph in which there is an edge between two nodes if and only if their distance is at most one. Connected dominating sets (CDS) have been proposed as a virtual backbone for routing in MANETs. Virtual backbone can be formed by nodes in a CDS of the corresponding UDG as suggested in [60][61][62][63] in a physically flat MANET. A virtual backbone or a spine plays a very important role in routing, where the number of nodes responsible for routing can be reduced to the number of nodes in the CDS. To reduce the communication overhead, to increase the convergence speed, and to simplify the connectivity management, it is desirable to find a minimum connected dominating set (MCDS) of a given set of nodes.

Centralized CDS construction algorithms are first proposed. A 10-approximation centralized algorithm for MCDS in UDG was first proposed in [64]. In 1998, Guha and Khuller proposed two greedy strategies for CDS construction in [65]. In the first strategy, CDSs are grown from one node outward. This algorithm yields a CDS of size at most $2(1+H(\Delta)).|OPT|$, where H is the harmonic function, Δ is the maximum degree of the graph, and OPT refers to an optimal solution that is the size of actual MCDS. Meanwhile, a Weakly-CDS is first constructed in the second strategy and then intermediate nodes are selected to create a CDS. The size of CDS created is at most $(3+\ln(\Delta)).|OPT|$.

However, distributed or localized CDS construction algorithms are more appropriate for MANETs. Das et al [60][61][62] proposed a distributed algorithm to form CDS by first finding an approximation to Minimum Dominating Set which is essentially the well-studied Set Cover

Problem. Let U denotes the dominating set output in the first stage. The second stage is to construct a spanning forest F where each tree component in F is a union of stars centred at the nodes in U. The stars are generated by letting each dominator picks up an arbitrary neighbour in U. The third stage expands the spanning forest F to a spanning tree T. All internal nodes in T form a CDS. It is a 3 H(Δ)-approximation of MCDS. The time complexity and message complexity of this algorithm can be as high as $\Theta(|V|^2)$.

Alzoubi *et al*. propose two distributed heuristics for constructing CDS in MANETs [66]. The first heuristic uses the ID-based approach for rank assignment. This approximation algorithm has a constant factor of 12. The second heuristic uses the level-based approach for rank assignment, and has a constant factor of 8. The message complexity of this approach (using an arbitrary spanning tree as a building block for the construction of a CDS) is $O(|V| \log|V|)$ and the time complexity is $O(|V|)$. Both algorithms consist of two phases. A maximal independent set (MIS) is first constructed based on a chosen rank definition which is induced by an arbitrary rooted spanning tree T. This spanning tree is constructed using the distributed leader-election algorithm in [67] with $O(|V|)$ time complexity and $O(|V|\log|V|)$ message complexity. It is obvious that the time and message complexity of their algorithms are dominated by this leader-election algorithm. After a leader is selected, the MIS construction procedure takes place. After the MIS is constructed, a dominating tree will be constructed and all nodes in the dominating tree forms a CDS. This algorithm achieves better performance in terms of the size of CDS. However, the message complexity is much higher than the optimal message complexity of $O(|V|)$. To achieve optimal message complexity, Alzoubi *et al*. propose another heuristic [68] that constructs a CDS in UDG without using a rooted spanning tree as in [66]. Initially all nodes are candidate. Whenever the ID of a node becomes the smallest among all of its one-hop neighbours, it will change its status to dominator. Then its candidate neighbours will become *dominatee*. After all nodes change status, each dominator identifies a path of at most three hops to another dominator with larger ID. The candidate nodes on this path become connectors. All dominators and connectors compose a CDS. The size of CDS constructed is at most 192.|OPT| + 48. It is clear that the performance in terms of CDS size has to be traded off for a lower message complexity. Moreover, the time complexity of this algorithm is still O(|*V*|), which may not be favourable in MANETs since the nodes may move during the CDS construction such that the resultant set of nodes is not a CDS.

Thus, the goal of the CDS construction is to have a constant time complexity for CDS construction. This goal has been realized by a truly localized algorithm, proposed by Wu and Li in [69], which adopts a prune-based CDS construction. This algorithm first finds a CDS and then prunes certain redundant nodes from the CDS to approximate MCDS. The initial CDS *U* consists of all nodes which have at least two non-adjacent neighbours. A node *u* in *U* is considered as locally redundant if it has either a neighbour in *U* with larger ID which dominates all other neighbours of *u*, or two adjacent neighbours with larger IDs which together dominates all other neighbours of *u*. This algorithm removes all locally redundant nodes from *U*. Approximation factor was unspecified in [69]. However, Wan et al. showed in [70] that this algorithm has poor performance over certain instances, in which the approximation factor is |*V*|/2. They also show that the message complexity and the time complexity of this algorithm can be as high as $O(|V|^2)$ and $O(|V|^3)$ respectively. Table 2.1 summarizes performance of above-mentioned distributed CDS construction algorithms. Most of the CDS construction algorithms attempt to approximate MCDS and form CDS that are as small as possible without taking other possible factors such as the cost of CDS and the stability of CDS into considerations. Taking network costs into account, Wang *et al.* [71] proposed a new algorithm to construct weighted CDS, whose size is guaranteed to be within a small constant factor with low cost. However, this algorithm also does not take the stability of the structure into consideration during the formation and maintenance of CDS.

Table 2.1 Comparison of three heuristics for distributed CDS construction

	Das et al	Wu & Li	Alzoubi et al
Approximation Factor	O (log n)	O (n)	8 - 12
Message Complexity	$O(n^2)$	$O(n^2)$	O (n log n)
Time Complexity	$O(n^2)$	$O(\Delta^2)$	O (n)

2.2.2.2 Maximum Weighted Independent Set Approach

Maximum Independent Set (MIS) is a special case of MWIS. MIS cluster construction algorithms can provide a virtual backbone by first constructing a MIS and then identifying a set of border nodes (a.k.a. gateway nodes) to be included in the special set that perform message routing and forwarding. Intuitively, MIS should have a small size as the nodes in an independent set are "sparsely" distributed with certain distance between any pair of nodes. Indeed, the size of any MIS in a Unit Disk Graph (UDG) is at most five times the size of the Minumum Dominating Set (MDS), as each node is adjacent to at most five independent nodes [64]. A tighter bound on the size of any maximal independent set in a UDG is proven in [66] to be at most 4|OPT| + 1, where |OPT| denotes the size of the optimal solution for the minimum dominating set problem on the UDG. Therefore, by computing a MIS, we may form a high quality dominating set and fulfil the independence property. Table 2.2 summarizes the main properties of different clustering algorithms presented in this discussion.

Table 2.2 Comparison of MWIS-based clustering algorithms

	Coverage	Maintenance	Clusterhead Election	Mobility-adaptive
L-ID	1-hop	Continuous Monitoring	Node ID	No
MCC	1-hop	Continuous Monitoring	Node degree	No
DMAC	1-hop	Continuous Monitoring	Combined metric of speed, node id and node degree	Partial
WCA	1-hop	Continuous Monitoring	Combined metric of speed, node id and node degree	Partial
MOBIC	1-hop	Continuous Monitoring	Aggregate Local Mobility	Yes
Max-min D-Cluster	*d*-hop	Periodical Reclustering	Node ID	No
(α,t)-Cluster	*d*-hop	Continuous Monitoring	Mobility profile	Yes

In prior work, heuristics proposed based on the greedy search for a MWIS in MANETs are based on: (1) node ID or (2) node connectivity degree to its neighbours. Linked Cluster Architecture (LCA) [46] is one of the earliest clustering algorithms for MANET, which uses node ID as clusterhead criterion. LCA was developed for packet radio networks and to be used with small networks of less than 100 nodes. LCA organizes nodes into clusters on the basis of node proximity. Each cluster has a clusterhead, and all nodes within a cluster are within direct transmission range of the clusterhead. Gateways are nodes that are located in the overlapping region between clusters. Two clusters communicate with each other via gateways. Pair of nodes can act as gateways if there are no nodes in the overlapping region. LCA was later revised in [72] to reduce the number of clusterheads. In the revised version of LCA, a node is a covered node if it is in the 1-hop neighbourhood of a node that has declared itself as clusterhead. A node declares itself to be a clusterhead if it has the lowest ID among the non-covered nodes in its 1-hop neighbourhood. This algorithm is then known as Lowest-ID algorithm.

Parekh [48] suggests using the degree of connectivity instead of the node ID in the clusterhead election. This algorithm is known as Maximum Connectivity Clustering (MCC). The node with maximum number of neighbours is elected as a clusterhead and any tie is broken by the unique node ID. The neighbours of a clusterhead become members of the cluster. Covered nodes will not participate in subsequent clusterhead election. This algorithm suffers from the variation of node degree due to the frequent changes of network topology. The variation in node degree will trigger frequent cluster re-organization and high signaling overhead will be incurred.

Basagni *et al.* then proposed a generalized algorithm, Generalized Clustering Algorithm (GCA) [73] which provides a more general way to express preferences through the choice of weights. A combined weight of node degree, mobility, and power can be set to account for multi-parameter optimization as suggested by Chatterjee et al. in Weighted Clustering Algorithm (WCA) [74]. Basagni also enhanced GCA in subsequent work [49][75][76], which is Distributed Clustering Algorithm (DCA) and Distributed Mobility-Adaptive Clustering Algorithm (DMAC). Basagni stated three ad hoc clustering properties [75] that should be fulfilled by every clustering algorithm:

i) Every ordinary node has at least a clusterhead as neighbour (*dominance* property).

ii) Every ordinary node affiliates with the neighbouring clusterhead that has the biggest weight.

iii) No two clusterheads can be neighbours (*independence* property).

The time complexity of DCA is bounded by a network parameter which depends on the topology of the network rather than its size. However, DCA is more suitable for quasi-static networks because all nodes are not allowed to move until end of the initialization phase, and the maintenance of DCA clusters is done periodically by rerunning the initialization phase. Meanwhile, DMAC improves DCA by relaxing the constraint that the nodes must remain still during cluster initialization and clusters the nodes based on the mobility of the network. The maintenance of cluster structure is adaptive to the mobility of nodes. Each node will react accordingly when it senses topology changes in its surrounding. Hence, it is claimed to be more suitable for any mobile environment. In DMAC, the initialization process of a node is similar to Lowest-ID and MCC. The role of a node is decided by its own weight and the weights of its one-hop neighbours. In order for proper execution, all nodes must know their own weight and role as well as the weight and role of each neighbour in a continuous fashion. As the nodes move around, they will observe the neighbourhood and react accordingly. E.g. if a node loses contact with its clusterhead, it will determine its role again as in the initialization process. Like GCA, DMAC adopts a weight-based mechanism in clusterhead election. Although the weight can be configured for specific applications, it is usually difficult to make it adaptive to the network condition at a specific instance of time. For example, in order to let the weight of a node represent its speed, every node in the network has to be aware of its own speed at every instance of time. Some nodes may not be equipped with the ability to know their own speed.

Basu et al. propose a weight-based clustering algorithm, MOBIC [52], which is similar to DMAC. Instead of node speed, MOBIC uses a new mobility metric, Aggregate Local Mobility (ALM), to elect a clusterhead. The ratio between the received power levels of successive transmissions between a pair of nodes is used to compute the relative mobility between neighboring nodes, which determines the ALM of each node. Therefore, the clusterhead election criterion is more adaptive towards network mobility.

All of the abovementioned algorithms may suffer from the changes in network topology which may cause clusterheads to be re-elected each time the cluster membership changes. For example, a node with ID lower than the clusterhead comes into cluster coverage and snatches the role of the clusterhead. These events trigger frequent changes of clusterheads and cluster reconfiguration, which can incur prohibitive overheads. Therefore, the Least Clusterhead Changes (LCC) mechanism [45] is designed to minimize these changes. LCC suggests that a clusterhead change only occurs when two clusterheads come within range of each other, or a node becomes disconnected from any cluster. Though clusterhead changes can be reduced by using LCC, it cannot be eliminated completely since the network topology is dynamic. When two clusterheads come into direct contact, one of the clusterheads will give up its role (this clusterhead is referred to "loser"). Some nodes in the loser cluster may not become members of the winner cluster. Therefore, one or more of those nodes must become a clusterhead. Such changes may propagate across the network, causing a rippling effect of clusterhead changes [55].

Apart from the instability in the cluster structure, these algorithms only form one-hop clusters in MANETs. Therefore, they are more suitable for dense MANETs in which most of the nodes are within direct transmission range of clusterheads. However, these algorithms may form a large number of clusters in relatively large and sparse MANETs and eventually lead to the same problem as in a flat architecture. Nocetti *et al.* [77] and Amis *et al*. [50] generalized the clustering heuristics so that an ordinary node can be at most *k* hops away from its clusterhead. Nocetti *et al.* proposed Connectivity-based K-Hop Clustering [77]. In this algorithm, a clusterhead is elected based on node degree as primary criterion and node ID as secondary criterion. Meanwhile, Amis *et al.* proposed Max-min *d*-Clustering [50] that uses node ID as the only criterion. Both algorithms allow more control and flexibility in the determination of clusterhead density. However, clusters are formed heuristically without taking node mobility and their mobility pattern into consideration. Moreover, the hop count parameter, *k* (or *d* in Max-min *d*-Clustering) is a pre-fixed parameter during the execution of algorithm. It is desirable to have the cluster diameter being able to adapt dynamically to some important network parameters, e.g. mobility patterns and network size. Banerjee *et al.* [78] proposed a clustering scheme to construct hierarchical control structure for multihop wireless networks by using certain geometric properties of wireless networks. However, this scheme is more suitable for stationary wireless networks. All of these multihop clustering algorithms require or assume knowledge of *k*-hop neighbours. Undoubtedly, this requirement imposes additional burden on MANETs which possess scarce bandwidth.

In a MANET consisting of individuals that exhibit uncoordinated mobility, clustering may incur a large number of re-clustering and thus a substantial amount of clustering overhead. Therefore, being able to cluster the nodes only when they show a certain coordinated moving pattern like group mobility is more appropriate than static clustering. This can avoid unnecessary cluster restructuring. McDonald and Znati [58] proposed an approach, namely, a (α,t)-clustering algorithm that adaptively changes its clustering criteria based on the current node mobility. This algorithm determines cluster membership according to a cluster's internal path availability between all cluster members over a certain time period. The (α,t)-Cluster Protocol presents a strategy for dynamically organizing the topology of an ad-hoc network. The cluster formation will be more likely to happen in networks with low rates of mobility. However, cluster size will be diminished when mobility rates become very high. Based on the (α,t)-Cluster framework, intra-cluster routing requires a pro-active strategy whereas inter-cluster routing is demand-based. Consequently, the framework specifies an adaptive-hybrid scheme whose balance is dynamically determined by node mobility rate. Random Walk Based Mobility Model [79] is used to determine the probability of path availability when links are subject to failure due to node mobility. The assumption of a specific mobility model may restrict the use of this scheme in networks that adopt other mobility models. Moreover, the movement generated by the Random Walk Based Mobility Model may not be realistic [80]. Other mobility patterns e.g. group mobility model, are not considered in this scheme. All cluster members must be aware of each member's existence and their mobility profile in order to evaluate path availability or link availability with respect to each member. Therefore, the amount of information it needs to maintain is at least O(m), where m is the number of members in a cluster. If cluster members are allowed to be k hops away from each other, this algorithm requires knowledge of its k-hop neighbours. In addition, nodes have to be aware of their instantaneous speed in order to determine the probability of path availability.

2.3 Network-layer Multicast Problem in MANETs

The standard IP multicast model has been introduced and described by Steven Deering [15] in 1988. The IP multicast model proposed by Deering is based on a notion of groups. Hosts that are interested in a particular application form a multicast group. Each multicast group is identified with a special class-D IP address. To receive data from a multicast group, hosts must join the group by contacting the routers they are attached to, using the Internet Group Management Protocol (IGMP) [81]. Once a host joins a group, it receives all data sent to the group address regardless of the senders' source address. Given that the scope of IGMP interaction is limited to a host and its attached router, another protocol is needed to coordinate the multicast routers (including the attached routers) throughout the Internet, so that multicast datagrams are routed to their final destinations. This latter functionality is accomplished by the network-layer multicast routing protocols, such as PIM, DVMRP, CBT and MOSPF. A multicast delivery infrastructure, which is usually a tree in IP multicasting context, is constructed and maintained by the multicast routing protocol. Most multicast routing trees are constructed using the Shortest Path (SP) algorithm or Minimum Spanning Tree (MST) algorithm.

In the context of MANET, there is no explicit protocol proposed to handle multicast group membership and most of the existing multicast routing protocols in MANETs assume a SP or MST algorithm to construct either a source-based or shared multicast delivery structure, which may be a tree or a mesh. A number of new multicast routing protocols are proposed in MANETs to address salient challenges due to the nature of MANETs. In the following subsections, greater details on previous work done in network-layer multicasting in MANETs will be discussed.

2.3.1 Multicast Group Management

In IP multicasting, the Internet Group Management Protocol (IGMP) operates between a host and its directly attached router. IGMP provides the means for a host to inform its attached router that an application running on the host intends to join a specific multicast group. In MANET, there is no explicit protocol proposed so far to handle the group membership. Most of the existing multicast protocols assume the source node or a special elected core node/multicast group leader to maintain the membership of multicast group. This assumption may increase the workload on the source node or the special node if the group size, i.e. the number of multicast receivers, is large.

2.3.2 Multicast Path Setup Algorithm

Each multicast routing protocol consists of two processes: (1) setup of the multicast delivery structure and (2) maintenance of this multicast delivery structure. The first process involves setting up paths that connect the source node and each multicast receiver. The union of these paths may appear as different kinds of forwarding infrastructure such as Shortest Path Tree (SPT), Minimum Spanning Tree (MST), minimal Steiner trees, acyclic meshes and so forth. The underlying forwarding infrastructure is protocol-specific because it largely depends on the underlying multicast path setup algorithm used by a particular protocol. In general, the multicast path setup process can be initiated either by the source node or the receiver node, i.e. source-initiated scheme or receiver-initiated scheme. Source-initiated multicast path setup begins with the dissemination of a request packet by the source node to indicate its intention to initiate a multicast session. Upon receiving this request packet, each multicast receiver will send a reply packet back to the source node using the path from which it arrives to indicate their intention to join the multicast session. Most existing MANET multicast routing protocols adopt shortest path algorithm in the source-initiated multicast path setup. Therefore, the delivery structure formed is usually optimal in terms of the path length between source and every receiver and thus optimal delay performance is expected. However, these trees are not optimal in terms of the overall number of forwarding nodes and thus, a higher data overhead is usually incurred. Source-initiated multicast path setup usually forms a source-based tree for each source in the multicast group. Therefore, it is more suitable for one-to-many multicast communications. Figure 2.1 shows an example of source-based multicast tree formed by the shortest path heuristic. This tree consists of four forwarding nodes and involves seven links.

On the other hand, receiver-initiated multicast path setup usually builds a shared tree e.g. MAODV, AMRoute and AMRIS. It requires a node to be the leader (core) of a multicast group session which is usually the first node that joins the multicast group (may not be the source node). Every multicast receiver first sends the join packet to the entire network in order to inform the source node about their presence. If there is no reply from other nodes, this node will assume the role of the leader node like the core in CBT. Otherwise, the first (nearest) node on the multicast delivery structure that receives this join packet will reply with an allow-to-join packet and a shortest tree link between the reacting node and the receiver node is formed. This tree is usually less optimal in terms of the path length (delay) between source and multicast destinations. However, it may be more optimal in terms of bandwidth consumption since the path setup heuristic is based on the minimum spanning tree algorithm and multiple source nodes in the same multicast groups can share a single tree. However, the cost of these shared trees is largely affected by the sequence of nodes joining the multicast session and therefore it is hard to predict the optimality of the delivery structure in terms of bandwidth consumption. Moreover, this approach incurs a substantial amount of control overhead if the group of multicast receivers is large. Figure 2.2 shows an example of shared tree based on the nearest tree link heuristic, in which the sequence of node addition is indicated by the number at the upper right corner of the relevant circles. In this case, the shared tree formed is not only sub-optimal in terms of <source, destination> path length, it is also sub-optimal in terms of bandwidth consumption. Four nodes are chosen as the forwarding nodes while six links are involved in this tree.

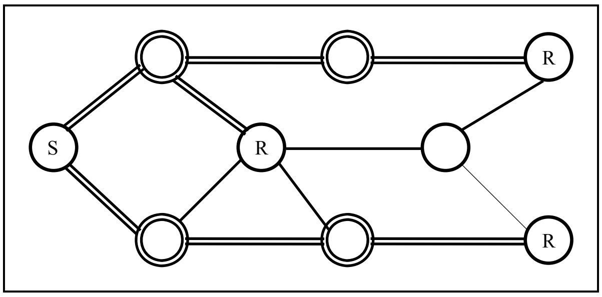


Figure 2.1 **Source-based shortest path tree (4 forwarding nodes, 7 links).**

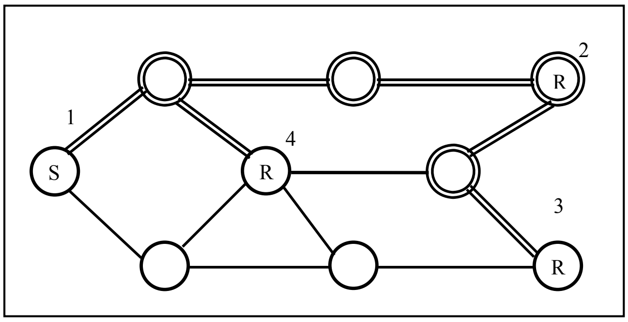


Figure 2.2 **Shared tree based on nearest tree link addition heuristic (4 forwarding nodes, 6 links).**

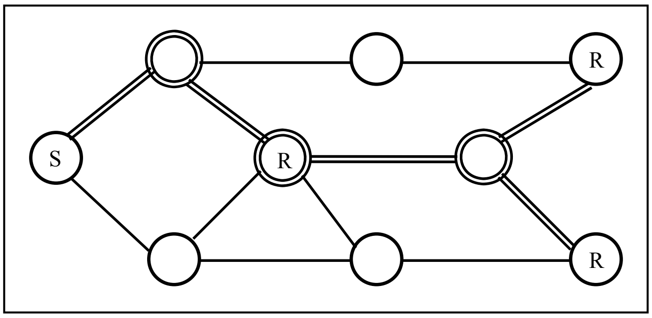


Figure 2.3 **Steiner tree (3 forwarding nodes, 5 links).**

The problem of finding a bandwidth-optimal multicast tree is well-known as the Steiner tree problem. This problem has been shown to be NP-complete by Karp [81][82] even when every link has the same cost. There are a number of heuristics proposed in the literature to approximate minimal Steiner tree in a centralized manner. For example, the Minimal Spanning Tree (MST) algorithm provides a 2-approximation [18]; as well as a 1.55-approximate algorithm proposed by Robins and Zelikovski [83]. Due to the distributed nature of MANETs, centralized algorithms are not directly applicable. Moreover, the set of multicast receivers in the multicast application in MANETs is usually dynamic and this piece of information may not be readily known by every node. Therefore, this translates the centralized offline Steiner problem into a distributed, online Steiner problem. Due to the fact that computing bandwidth-optimal multicast structure is computationally infeasible in MANETs, little work has been done in this area. However, there has been an extensive range of work done in other fields to address centralized online Steiner problems.

The centralized online Steiner tree problem has been well-studied in the literature [84][85]. In the centralized online Steiner problem, the input to the algorithm consists of a graph G and a series of vertices $v_1, v_2, \ldots, v_n$ which is revealed one at a time. At each vertex request v_i, the algorithm must compute T_i, a Steiner tree that spans $v_1, v_2, \ldots, v_i$ with the constraint $T_{i-1} \subseteq T_i$. That is, the tree is constructed incrementally at every request. A common approximation to the Steiner tree is a greedy Steiner tree; a greedy algorithm chooses to incur the minimum incremental cost at each request. Imase and Waxman [86] investigated the problem of constructing greedy Steiner tree that considers both vertex addition and removal requests. They have proven that the competitive ratio to construct such a greedy Steiner tree is $\lceil \log n \rceil$. Since this greedy Steiner tree algorithm is centralized in nature, it cannot be applied directly to MANETs.

Given the limitations in MANETs and the complexity to compute these trees in a distributed manner, the shortest path algorithm is more commonly in use because it can be easily computed in polynomial time. Figure 2.3 shows a Steiner tree which is optimal in terms of bandwidth consumption. This tree consists of 3 forwarding nodes and 5 links.

2.3.3 Multicast Routing Protocols

In this section, existing multicast routing protocols for MANETs are categorized into six different categories based on their delivery infrastructure as shown in Figure 2.4. Table 2.3 compares various properties of tree-based and adaptive multicast routing protocols whereas

Table 2.4 compares those of flooding approach, mesh-based and hybrid multicast routing protocols. The aspects that are taken into account include packet forwarding infrastructure, stand-alone capability, route discovery mechanism, periodic overheads, optimality of route, and scalability in terms of the number of senders, loop formation and group member leaving approach. In a later section, a comparison between tree-based and mesh-based multicast routing protocols is presented.

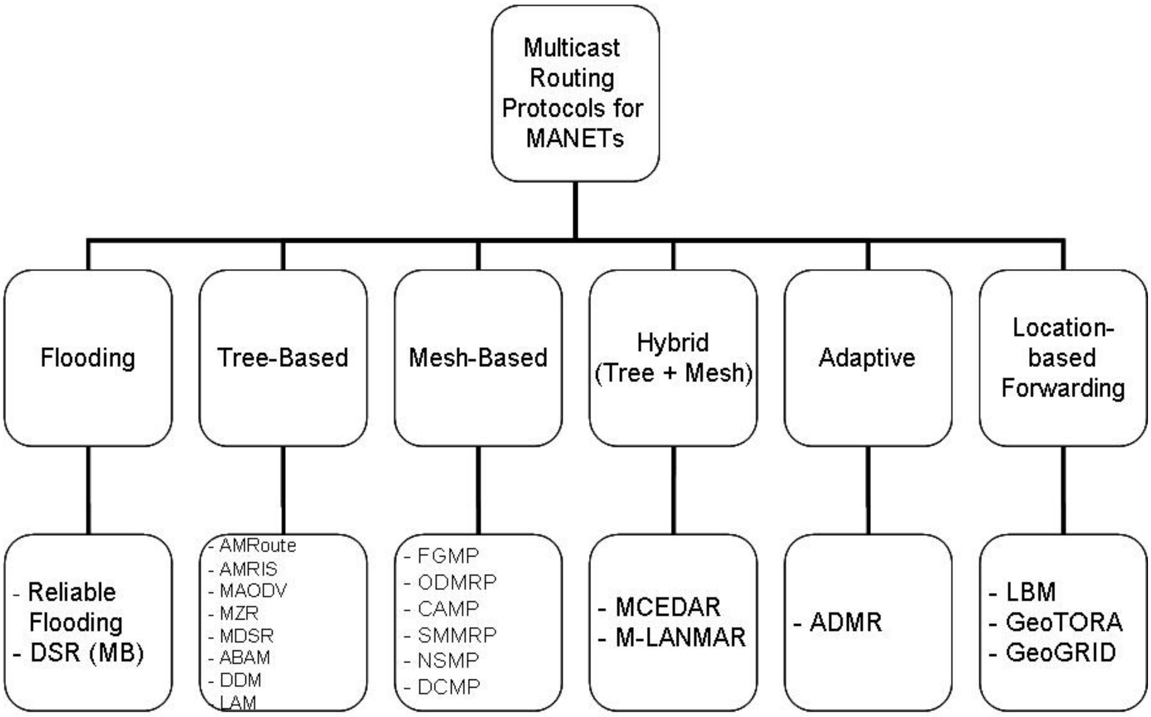


Figure 2.4 Categorization of multicast routing protocols for MANETs.

2.3.3.1 Flooding Protocols

Flooding is proposed in [27] and [87] as the most straight-forward and robust way to perform multicast routing in highly dynamic, fast-moving MANETs. Flooding is the easiest way to perform multicasting since it eliminates the need to build and maintain an explicit multicast delivery infrastructure. When a source wishes to send a multicast packet, it broadcast the packet to its neighbors. Upon receiving the multicast packet, the node determines whether the packet has been received before. If not, it rebroadcasts the packet to its neighbors.

Otherwise, the packet is discarded. This process is repeated until the packet is flooded throughout the network.

However, flooding is inefficient since the network would be congested by multiple copies of the same data packet. As pointed out in [88], flooding in a MANET may cause serious contention, collision and redundancy, resulting in severe packet loss. The problem becomes more serious when the network becomes larger. In other words, flooding is not a scalable approach. Attempts were made to improve the performance of flooding by restricting the forwarding space, such as the forwarding group concept in FGMP [89] and controlled flooding in Simple Multicast and Broadcast Protocol [90] to distribute data in small networks with very high degree of mobility. Key characteristics of flooding protocols are compared with the mesh-based, and hybrid protocols in Table 2.4.

2.3.3.2 Tree-based Protocols

The tree-based approach is a well-established multicast mechanism used in wired networks. A single tree is deployed as the packet delivery structure with one particular node, either source or core, acting as the root. Most proposed schemes use either source-based or shared tree to facilitate packet forwarding. The former one usually constructs a multicast tree per source node based on the shortest-path algorithm. The latter one usually constructs a multicast tree per multicast group based on the MST algorithm. Tree-based protocols are generally more efficient in terms of data transmission but less robust than flooding mechanism and mesh-based protocols. The following discussion is based on Table 2.3 that shows the various characteristics of eight tree-based multicast routing protocols for MANETs, as follows:

i. *Associativity-Based Ad hoc Multicast (ABAM)*: ABAM [91] is an on-demand multicast routing protocol for MANETs that uses the association stability concept in Associativity-Based Routing (ABR) [92] to establish a stable source-based multicast tree for every multicast session. The association stability concept leads to less tree reconfigurations and therefore less communication overheads. Like other tree-based protocols, ABAM provides only one path while demanding larger storage resources keeping track of multiple source-based trees. Scalability is another problem as its multicast routing table grows linearly with the number of senders.

ii. *Multicast Zone Routing (MZR)*: MZR [93] builds a multicast tree for each source-group pair in every multicast session, using the zone routing mechanism, which is first introduced in the Zone Routing Protocol (ZRP) [94]. Inside the zone, proactive routing is used where every node stores a zone routing table and periodically exchanges this table with all other nodes within its zone. On the other hand, on-demand routing is employed when the multicast source intends to send data across the zones. The operation of MZR protocol can be divided into two parts: (1) zone construction and maintenance, and (2) multicast tree creation, maintenance and deletion. In line with the zone routing concept, every node in the MANET constructs a zone around itself with a pre-configured zone radius. To construct and maintain their zone, each node maintains a zone routing table as well as a neighbor table using a simplified distance vector algorithm. The zone routing table is kept up-to-date through periodic advertisements. The two-stage multicast tree creation, as illustrated in Figure 2.5 is initiated by a multicast source when it intends to send data packets to multicast receivers. MZR aims to reduce routing overhead by preventing unnecessary broadcasts of tree discovery and recovery packets to the entire network. However, this two-stage tree creation introduces extra latency before the first multicast packet can be sent to intended receivers. In terms of protocol scalability, this approach seems promising but the introduction of excessive delays and overhead in large-scale and high density networks are inevitable as the zones are heavily overlapped. Besides, the selection of zone radius can greatly influence the performance of the protocol. Optimizing a general parameter for MZR is not a trivial task since different networks have different properties. Furthermore, every node in the network assumes the same zone radius. This is a less adaptive approach. MZR would be more beneficial if the branch reconstruction can be completed within the zone where link breakage is detected. Bandwidth is conserved as the tree repair is localized. Conversely, if multicast receivers are sparsely distributed throughout the network, the repair operation may need to flood the entire network with recovery packets and the latency is inevitable.

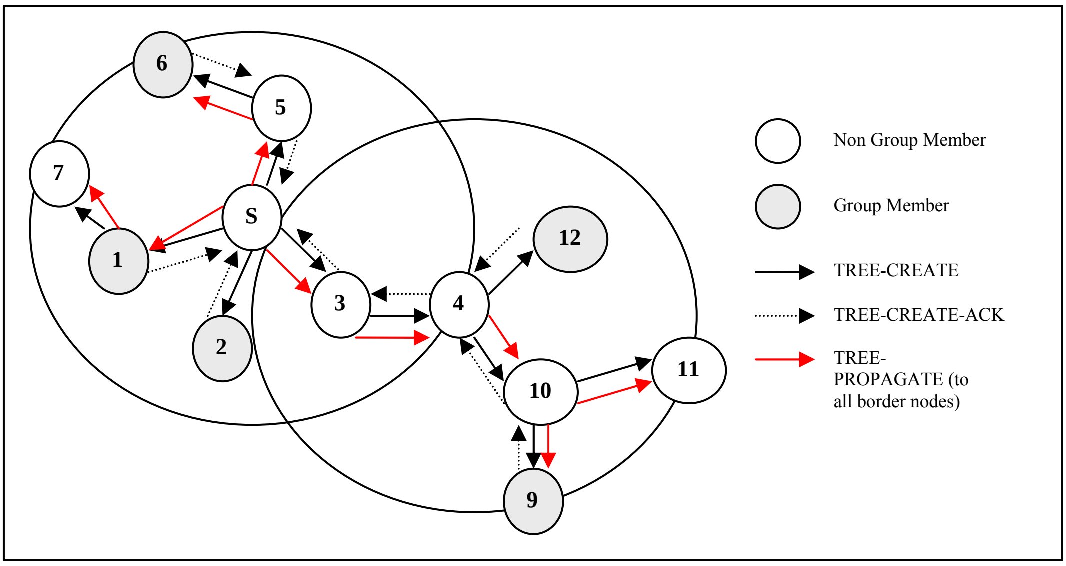


Figure 2.5 Two-stage multicast tree creation in MZR.

iii. *Differential Destination Multicast (DDM)*: DDM [95] is not a general purpose multicast protocol as it is designed to handle multicast groups of limited sizes, and it relies on an underlying unicast routing protocol for all routing information. For each multicast session, identified by a source-group pair, a multicast delivery tree rooted at the source node is implicitly built. Therefore, DDM is an on-demand and stateless protocol where there is no need to store multicast routing table in any participating nodes. This avoids loading the network with pure signaling traffic or control overhead when there is no data traffic. However, the size of data packet grows linearly with the number of multicast group members. DDM algorithm may also need some modifications to incorporate certain unicast routing protocols for correct and effective routing. Due to this reliance, the performance of DDM is greatly dependent on the performance of the underlying unicast routing protocol.

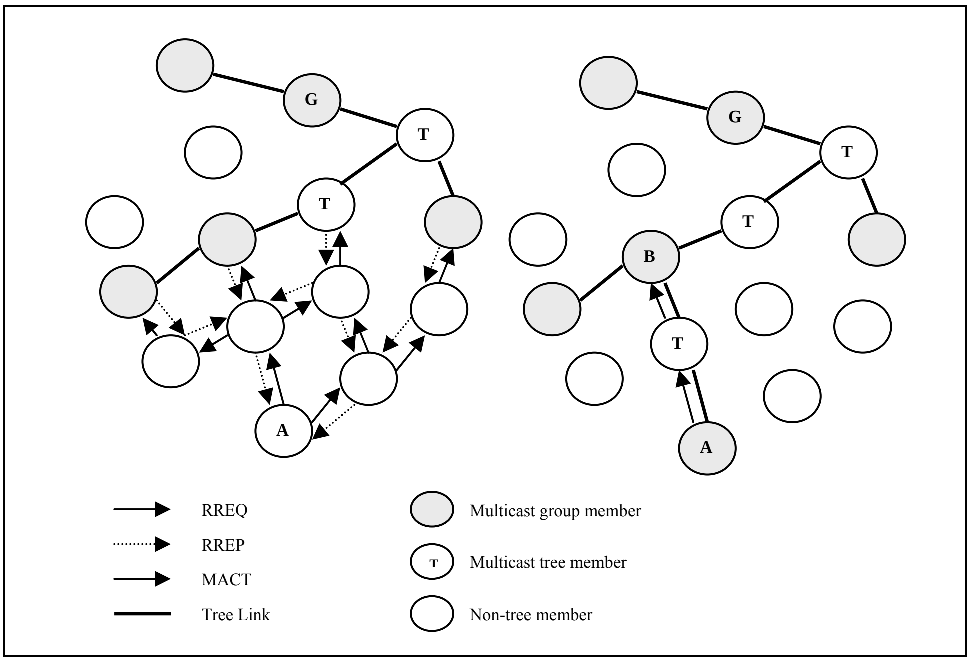


Figure 2.6 Multicast join operation in MAODV.

iv. *Multicast Ad hoc On-Demand Vector (MAODV)*: Being a simple extension of the unicast Ad-hoc On-Demand Vector (AODV) [96] protocol, the MAODV [97] builds routes with a small amount of overhead from control messages without piggyback any additional routing information on data packets. It also ensures efficient repair after link breakages or network partitioning. Its tight integration with AODV allows it to inherit the advantages, e.g. loop-free routes, efficient routing, etc [98]. It also conserves bandwidth as it constructs a shared-tree for each multicast group. However, like other tree-based multicast protocol, MAODV only provides a single route to destination. Although it incurs small amount of overhead during route establishment, MAODV requires periodic neighbor sensing for local connectivity management as well as periodic broadcasts of group control message to maintain group connectivity. The group leader for every multicast group is a central point of failure. Network mobility tends to invoke more switching of group leaders, incurring more overhead, delay and data loss during packet transmission. Like AODV, MAODV initiates an on-demand route request/route reply discovery cycle. A node originates a Route Request (RREQ) message when it wishes to join a multicast group, or when it has data to send to a multicast group but it does not have a route to that group. This RREQ message is then rebroadcasted by all the intermediate nodes until it reaches an on-tree node (nodes that are part of the multicast tree for that particular multicast group). This on-tree node will reply a Request Response (RREP) message by unicast along the reverse path to the RREQ initiator. The RREQ initiator node may get more than one RREPs during the route discovery interval. During this interval, it collects RREPs and selects the best route (route with the greatest sequence number and the smallest hop count to the multicast tree). At the end of this interval, it activates the selected next hop and unicasts a multicast activation (MACT) message to the selected next hop. Upon receiving MACT, the node becomes one of the tree members. If it is already tree member, it stops propagating MACT. Otherwise, it selects the best route to follow and sends MACT to the best next hop selected. The process continues until the node that originated the RREP is reached. A multicast join operation is illustrated in Figure 2.6.

v. *Ad hoc Multicast Routing Protocol (AMRoute)*: AMRoute [99] is a shared tree based multicast routing protocol that constructs single multicast tree for each multicast group regardless of the number of source. Using single shared tree per group improves its scalability with respect to the number of multicast senders. The important core nodes are not central point of failure, and multicast operation can proceed even without core nodes in a particular mesh segment. AMRoute heavily relies on an underlying unicast protocol to handle topology changes by creating bidirectional tunnels between group members. This is advantageous in that intermediate routers need not run any multicast protocol and overhead is confined to multicast group members only, but results in inefficient bandwidth usage and increased delay. The use of tunnels as tree links also implies that the tree structure does not need to change even in case of a dynamic network topology, which reduces signaling traffic and data loss [100]. However, when network topology changes in high speed, the links are more likely to provide only unidirectional connectivity and may lead to significant packet loss. Besides, transient loops may form during the transition from old to new tree, causing serious congestion [101] and low throughput.

vi. *Ad-hoc Multicast Routing Protocol utilizing Increasing id-numberS (AMRIS)*: AMRIS [102] is another shared tree based multicast routing protocol. It assigns a unique multicast id-number (*msm-id*) to each member in a multicast group, to handle the group membership and adapt rapidly to topology changes in MANETs. The logical and sparse ordering of the tree members using id-numbers facilitates quick local repair. Each node on the tree maintains an up-to-date neighbor status table but not global state. Unfortunately, periodic beaconing used in AMRIS may cause congestion and data packet collisions, as well as bandwidth and power wastage if the group is idle. AMRIS also assumes that multicast sessions are long-lived, and hence sacrifices route discovery latency to route recovery latency. If links break, repair is not done immediately as the breakage can only be detected after the predefined interval of time, during which packets may be dropped. Being shared-tree based, AMRIS is scalable in terms of the number of senders but the single routes between member nodes reduce the robustness of the protocol. Route optimality is also not guaranteed.

vii. *Lightweight Adaptive Multicast (LAM)*: LAM protocol [103] is a shared-tree based protocol, which sits on top of an on-demand unicast routing protocol, Temporal Ordered Routing Algorithm (TORA) [104]. It is tightly coupled with TORA to achieve both efficiency and simplicity. LAM is lightweight in terms of control overhead as TORA is in charge of maintaining link connectivity as part of its unicast routing operation. No additional overhead is introduced by LAM if topology remains stable. It sacrifices protocol portability to take advantage of TORA's route discovery and maintenance ability. Timing is also important in TORA. Therefore, LAM requires all nodes to be equipped with synchronized clocks (via an external time source such as the Global Positioning System). Another main problem of LAM is the reliance on a single node as core which may become a bottleneck and central point of failure that could paralyze the operation of the entire multicast group.

viii. *Multicast Dynamic Source Routing (MDSR)*: A multicast extension of Dynamic Source Routing, which is MDSR is proposed in [105]. It builds a minimal spanning trees based on all multicast routes discovered in DSR [106] route discovery process. The computed multicast tree is piggybacked in the header of data packet for routing purpose. High node mobility will trigger frequent re-computation of minimal spanning trees. Thus, this scheme incurs high overhead as well as low packet delivery ratio under highly mobile environment. It is also less scalable as the size of packet header grows with the size of multicast group.

2.3.3.3 Mesh-based Protocols

Unlike tree-based protocols, mesh-based multicast routing protocols deploy a set of nodes in the delivery structure, which will forward every incoming packet belonging to its multicast group. As a result, mesh-based protocols may have multiple paths for a single source-receiver pair and therefore more reliable than tree-based protocols. The discussion in the following section is based on Table 2.4. The followings are the six mesh-based protocols proposed in the literature:

i. *Forwarding Group Multicast Protocol* (FGMP): FGMP [89] is a multicast routing protocol for MANETs that builds multicast mesh instead of multicast tree. FGMP assigns a group of nodes as forwarding nodes for forwarding multicast packets. The forwarding group concept provides multiple routes to receivers and this increases the robustness of the protocol. However, periodic broadcast of membership advertisements incurs significant overhead and becomes a pure waste of bandwidth and power when there is no data traffic. Moreover, the size of the forwarding table grows linearly with the number of senders or receivers. As the mesh becomes thicker, multicast data packets may be flooded to nearly the entire network. FGMP needs to maintain a routing table itself or it may use routing information from other unicast routing protocols. To set up the forwarding group, two schemes are proposed in FGMP: Receiver Advertising (FGMP-RA) and Sender Advertising (FGMP-SA). Both schemes are similar except the node that periodically floods membership advertisement. In FGMP-RA, multicast receivers periodically flood its member information. When a sender receives these advertisements, it updates its member table. The sender will then create forwarding table, FW that contains next hop list. Next hop information is obtained from routing table that independently computed by the underlying unicast routing protocol. FW is broadcast and only neighbors which are included in the next hop list react to the incoming FW by constructing their own FW, enabling their forwarding flags and refreshing their forwarding timers. The broadcast of FW continues until all receivers are reached. The process is similar in FGMP-SA except the advertisements are periodically broadcast by the senders. Figure 2.7 is an example of a forwarding group set up to connect all receivers and senders in a mesh structure.

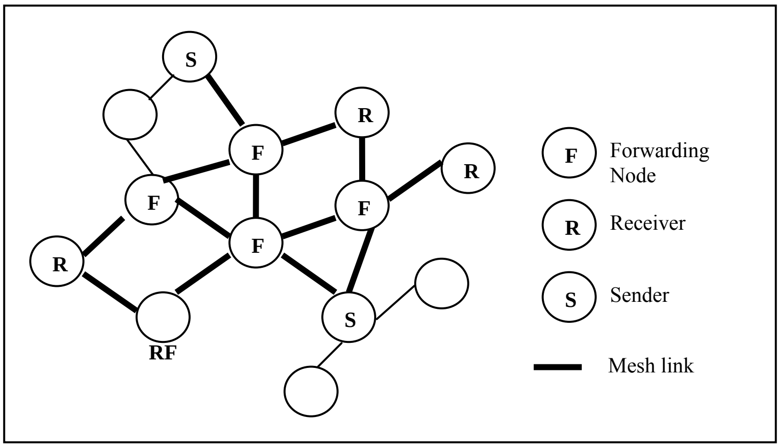


Figure 2.7 Forwarding nodes forming a mesh in FGMP.

ii. *On-Demand Multicast Routing Protocol (ODMRP)*: ODMRP [107], on the other hand, uses the forwarding group concept proposed in FGMP to dynamically build multicast mesh on-demand. ODMRP also has unicast capability. Like FGMP, the redundant routes provided in ODMRP can increase the packet delivery ratio but comes at the cost of additional overhead and load on the network. When the number of forwarding nodes increases and approaches the number of network nodes, ODMRP actually operates like pure flooding. As ODMRP builds per-source meshes, the thickness of meshes grows with the number of senders, making it less scalable in terms of the number of senders in a multicast group. How frequently a periodic flooding of control packets should be triggered becomes the main consideration in system design. Frequent flooding of control packets reduce the latency of link breakage discovery and thus reduce packet loss. However, a significant amount of control overhead is incurred. ODMRP forms the multicast mesh on-demand when a multicast source has data to send. ODMRP consists of two phases: request phase and reply phase. During the request phase, the multicast source will periodically flood a JOIN_REQ packet to: (1) refresh membership information and (2) maintain routing information. When a neighbor receives a non-duplicate JOIN_REQ, it stores the address of the packet sender as upstream node before it rebroadcasts the packet. Therefore, a backward path is implicitly created for routing the JOIN TABLE back to the source node during reply phase. When a multicast receiver receives the JOIN_REQ packet, the receiver creates and broadcasts a JOIN TABLE to its neighbors as reply. When a node receives a JOIN TABLE and realizes that it is on the path to the source, it will set the forwarding group flag and generate its own JOIN TABLE before broadcasting it to its neighbors. The JOIN TABLE is thus propagated by each forwarding group member back to the multicast source via the shortest path. This process constructs the routes from source to receivers and thus builds a shortest-path mesh (forwarding group), which consists of all forwarding nodes. The mesh structure is periodically refreshed through the global flooding of JOIN_REQ throughout the network. Multicast data packets are routed to receivers using the same forwarding mechanism as in FGMP.

iii. *Core-Assisted Multicast Protocol (CAMP)*: CAMP [108] builds a shared multicast mesh using routing information from a unicast routing protocol and the mesh consists of all reverse shortest paths from receivers to sources for each multicast group. In addition, CAMP ensures all reverse shortest paths are parts of the mesh to guarantee the route optimality. CAMP avoids the need to flood the entire network with control or data packets by using multiple core nodes, making it more scalable in terms of the number of multicast groups as well as the number of senders or receivers. The performance of CAMP also depends on the performance of the underlying unicast routing protocol especially its efficiency in link break discovery and recovery. Inefficiency in link break discovery and repair becomes more prominent when the node mobility increases. Control traffic also grows significantly in the presence of mobility.

iv. *Scalable Multi-source Multicast Routing Protocol (SMMRP)*: SMMRP [109] is a mesh-based multicast routing protocol, which forms a packet delivery mesh with a subset of the per source trees instead of a single tree or the entire set of per source trees as in ODMRP. Selecting a good proportion of core sources over sources is a crucial yet difficult decision in the design of SMMRP. The assumption of having a server as central administrator may not always hold in MANETs. If a static server is used throughout the entire network session, the server may become unreachable as network topology is dynamic and network partitioning may occur. This situation results in parts of the network components not having access to the server.

v. *Neighbour Supporting Multicast Protocol (NSMP)*: NSMP [110] is a multicast routing protocol that adopts mesh delivery structure to enhance its resilience against mobility. However, NSMP tries to restrict the size of the mesh structure in order to achieve multicast efficiency. Similar to ODMRP in some aspects, it attempts to achieve the improvement over ODMRP by localizing control messages to a small set of mesh nodes and neighbor nodes, and minimizing the frequency of network-wide flooding. NSMP tries to strike a balance between multicast efficiency and mesh robustness by restricting the size of mesh through reusing the forwarding nodes whenever possible.

vi. *Dynamic Core-based Multicast routing Protocol (DCMP)*: DCMP [111] builds and maintains a shared mesh, i.e. a mesh which is formed by a group of core based trees. The key concept is not to build trees based on all sources. Instead, DCMP assigns some sources to be active cores and these nodes forward data packets for passive nodes that are assigned to them. DCMP uses forwarding group concept and constructs the route in a similar way as in ODMRP. However, ODMRP maintains a set of source-based trees to form the mesh. The use of core-based trees reduces both control and data overhead, and also improves the scalability of the protocol. In other words, DCMP reduces the thickness of the mesh to achieve efficiency at the cost of reduced packet delivery ratio.

2.3.3.4 Hybrid/Adaptive/Hierarchical Protocols

The dynamically changing topology in MANETs presents a great challenge in designing a protocol that works well under most if not all conditions. An ideal multicast routing protocol should be able to give the best performance under different kinds of topologies that change over time during the multicast sessions. In this section, protocols that attempt to address both efficiency and robustness using hybrid approaches that combines the advantages of both the tree-based and mesh-based delivery structures, as well as those that can switch their multicast strategies based on the changes in the networks, are discussed as follows:

i. *Multicast Core-Extraction Distributed Ad-hoc Routing (MCEDAR)*: MCEDAR [28] is a multicast extension to the CEDAR [112] architecture that combines both the tree-based and mesh-based structures to exploit the efficiency of the tree-based forwarding protocols and the robustness of mesh-based protocols. A mesh is formed as the underlying infrastructure to ensure the robustness of protocol. Not every link break triggers a reconfiguration of the infrastructure and data packets can still be delivered to the receivers via other paths. An implicit source-based forwarding tree is created to ensure the data packets are forwarded via the shortest path. The delivery efficiency of MCEDAR approximates the efficiency offered by tree-based protocols in general. It relies on core broadcast instead of global broadcast, which may incur less network load. However, maintaining the parent-child hierarchy in the graph using a global ordering of group member may not be easy. Any change in topology may trigger a series of cascading changes in the entire graph. Furthermore, MCEDAR depends greatly on the performance of core nodes. As the set of core nodes is an approximation to the minimum dominating set, the failure of any core node will cause those nodes that are under its domination to lose connectivity with the group until they have found themselves a new dominator.

ii. *Multicast-enabled LANMARk ad hoc routing (M-LANMAR)*: M-LANMAR [14] applies a new multicast paradigm for large-scale MANETs, namely team multicast. The authors proposed to exploit team motion affinity. The approach assumes that the teams are predefined and do not change over the entire experiment. However, dynamic recognition of motion affinity among nodes in order to form groups can be used in M-LANMAR too. Multicasting is done between source and teams, not individual members. Multicast packets are sent to teams instead of individuals. M-LANMAR builds tunnels from multicast sources to each landmark of the subscribed team and restricted flooding within the motion group. A landmark for each team is first elected for a subnet as in Landmark ad hoc routing (LANMAR) protocol [113]. There are two complementary routing schemes as in LANMAR. First, it uses a myopic proactive routing scheme, operating within a limited scope centered at each node. Secondly, it uses a long haul distance vector routing that propagates the elected landmark of each subnet and the path to it into the whole network. Using this landmark updates, a team maintains its membership to multicast group. Membership is constantly refreshed, as each landmark includes subscribed multicast addresses to all outgoing landmark update packets. However, M-LANMAR suffers from the fact that not all team members are interested in multicast communication. If the team size increases, the overhead of limited scoped proactive routing will grow too. The election of landmarks is not specified clearly. Moreover, global broadcast packet for landmark maintenance may generate overhead in large-scale MANETs which, in turns, impedes protocol scalability.

iii. *Adaptive Demand-driven Multicast Routing (ADMR)*: ADMR [114] adapts by temporarily switching to flooding when the mobility of the network is too high for efficient multicasting. As a result, overhead incurred by ADMR scale gracefully with group size and increased mobility. In the multicast mode, source-based trees are created to forward multicast data packets. Each multicast data packet is forwarded from the sender to receivers using the shortest delay path. ADMR monitors the traffic pattern of the multicast source application and uses the information to efficiently detect link breaks and expired routing states that belong to inactive multicast groups. ADMR does not use periodic network-wide flooding of control packets, periodic neighbor sensing, or periodic routing table exchanging, and it does not require core. Therefore, no control overhead will be generated if the multicast group is idle. However, packet loss in MANETs may be caused by either mobility or collisions. Hence, it is possible that the routing layer may misjudge a link break due to packet loss caused by collisions. ADMR may switch to flooding unnecessarily and aggravate the congestion in MANETs.

iv. *Hierarchical Differential Destination Multicast (H-DDM)*: Another hierarchical multicast routing protocol proposed by Gui and Mohapatra in [115] is known as Hierarchical DDM (HDDM). The idea of HDDM is to extend the scalability of the DDM protocol which is known to be feasible only in supporting small multicast groups. To increase the scalability, HDDM divides the network into different sub-groups by electing a set of suitable sub-roots that are responsible for forwarding multicast packets. After the group division is completed, a source node will forward multicast packets to each sub-root using the DDM protocol. Each sub-root that receives the packet will disseminate the packets to its respective sub-group members using DDM as well. The effectiveness of HDDM is highly dependent on the performance of the underlying unicast routing protocol since HDDM acquires routes to the members listed in the DDM header from the existing unicast routing agent. The forwarding efficiency of HDDM is not optimized since it is based on the unicast routing protocol that finds the next-hop to destinations using the shortest path algorithm. Furthermore, HDDM also requires the source node to have a complete list of group members, initiate the partitioning process and maintain the optimal partitions during the multicast communication. All group members have to participate in the initialization of multicast session.

2.3.3.5 Location-based Protocols

Location-based forwarding or geocasting is a routing method that takes the location information of each node into considerations when making the routing decision. Geocasting

[116] is a variant of the conventional multicasting problem as a geocast is delivered to the set of nodes within a specified geographical area. Therefore, the set of nodes are multicast group members while the specified area is similar to the multicast group in the conventional context. Geocasting can be useful when a message is targeted at a group of receivers within a specified geographical region. There are several geocasting protocols proposed for MANETs, such as Location-Based Multicast (LBM) [117], GeoTORA [118] and GeoGRID [119]. The discussion here is summarized in Table 2.6, which compares the characteristics of the following location-based protocols:

i. *Location-Based Multicast (LBM)*: LBM uses the forwarding zone concept to restrict the degree of multicast flooding in the network and thus reduce multicast overhead. The coordinates of the forwarding zone, smallest rectangle that includes the current location of sender and multicast region, will be piggybacked on multicast packets. No periodic control overhead is needed as the multicast packet delivery is done through flooding.

ii. *GeoTORA*: GeoTORA, on the other hand, is proposed to enhance LBM. GeoTORA incorporates TORA to restrict the flooding within a small region. GeoTORA performs geocasting in two phases. Firstly, the data packet is anycast through TORA to one of the geocast group members. Once a packet is delivered to one node in the geocast group, that node initiates local flooding of the packet. GeoTORA improves LBM by further reducing data overhead. However, additional overhead to maintain anycast routing information through TORA must be taken into account.

iii. *GeoGRID*: GeoGRID uses the GRID structure [120] to perform geocasting in MANETs. In GeoGRID, the geographic area of the MANET is partitioned into two-dimensional logical square grids. In each grid, one mobile host (if any) will be elected as the grid leader. Geocasting is then performed in a grid-by-grid manner through grid leaders. Like LBM and GeoTORA, the information of destination region is piggybacked on each multicast data packet. Instead of allowing each host in destination region to forward data, GeoGRID only allows grid leaders that are within the destination region to take this responsibility. Therefore, GeoGRID can eliminate redundant transmission of geocasting messages and maintain a high arrival rate of geocasting messages at the same time. However, choice of grid dimensions is a difficult decision in the design of this protocol.

2.3.3.6 Tree-based vs. Mesh-based Multicast Routing Protocol

As most of the multicast routing protocols are using either tree-based or mesh-based approach for their multicast delivery infrastructures, a comparison is done between these two approaches. Table 2.5 shows the comparison of these two approaches in addressing multicast routing in mobile ad hoc networks.

Tree-based protocols generally provide a more efficient approach for multicasting as data packets are duplicated at tree forks only. The minimum number of copies per packet is used to disseminate multicast packet to all the receivers. However, these protocols are more vulnerable to mobility since there is only one path between any pair of nodes. When the routes break due to mobility, packets must be buffered or dropped until the tree is repaired. Every single link failure requires reconfiguration of the multicast tree. This incurs substantial control overhead if the relative mobility is high and the underlying topology of MANETs changes rapidly.

In the contrary to tree-based protocols, mesh-based protocols provide redundant paths for packet delivery. Therefore it is usually more robust against mobility. It is shown in [101] and [121] that mesh-based schemes (e.g. flooding and ODMRP) perform considerably better than tree-based scheme (e.g. MAODV, AMRIS, AMRoute) in the presence of mobility. The number of forwarding nodes in the mesh determines the robustness of the protocol. However, a higher number of forwarding nodes incurs higher overhead. Meshes that consist of source-based trees become thicker when the number of sources increases. Therefore, these protocols are less scalable with respect to the number of senders in a multicast group.

2.3.4 Survey Summary and Open Issues

Multicasting is a more efficient method of supporting group communication especially in resource-limited mobile ad hoc networks. Since the direct extension of the wired multicast routing protocols to MANETs appears unfeasible, much work has been done to design new multicast routing protocols for MANETs.

Generally, flooding is the most reliable approach to forward multicast packets in MANETs as its packet delivery ratio outperformed other protocols in performance comparison done in [101]. However, it is not scalable with the size of multicast group as well as the network. Its primary overhead is the multiple copies of the same packet circulating in the network. Though attractive in terms of its robustness againstmobility, flooding may not be suitable for multicast routing in MANETs. The tree-based concept that prevails in wired networks is then introduced for multicasting in MANETs. However, tree-based protocols are not robust because they provide only a single path between two nodes, making them vulnerable to node mobility. For example, mesh-based protocols like ODMRP [122] and CAMP [123] exhibited better performance under high mobility compared to tree-based protocols such as AMRoute [99] and AMRIS [102]. In general, mesh-based protocols offer better robustness against mobility at the cost of higher data overhead. Thus, it is less scalable in terms of traffic load since a large amount of data overhead may cause the network to saturate faster. Though AMRoute and AMRIS show a more stable performance with the growth of the senders, these protocols are less robust. Meanwhile, hybrid and adaptive protocols have been proposed to take advantage of both tree and mesh structures.

Different multicast approaches should be deployed for different kinds of network environment, but the dynamic behaviour of MANET increases the complexity of this problem. Designing an adaptive multicast routing protocol that is adaptive to different MANET environments, e.g. [114], [124], and [125], remains an open problem. Despite the numerous protocols available, a satisfactory solution for MANETs is still not evident and there remain a number of issues and open problems that require further investigation and research, e.g. reliability, scalability, security, QoS support, and power consumption.

In this research, the scalability issue was given emphasis. Most of the proposed multicast protocols for MANETs are designed with small network and flat topology in mind (Previous protocols were simulated using network scenarios of 50-100 nodes and the average size of geographical area used in simulations is 1 km^2). This observation instigates the question of how large can a MANET grow before existing multicast protocols fail to meet expectations. As defined earlier, protocol scalability is the ability of the protocol to support the continuous increase of the network parameters (such as network density, mobility rate, data generation rate) without degrading network performance [32]. This motivates the need to design a multicast routing protocol for MANETs with increased scalability in terms of a set of network parameters such as network density, mobility rate and data generation rate. Apart from scalability, the multicast routing protocol should maintain its efficiency and robustness. Therefore, the main research question for this thesis is formulated as follows:

Can we design a multicast routing protocol for large MANETs, which is efficient in terms of network bandwidth consumption, robust against mobility, and adaptive to network condition?

As discussed in the previous sections, clustering is a one of the common approaches used to increase scalability in communication networks. Take flooding as instance; Taek and Gerla [54] suggest an approach to reduce the flooding overhead by using a minimal subset of forwarding nodes which is sufficient to deliver the packet to every other node in the system. They claim that by using one-hop clustering structure for flooding, one can achieve about 80% of overhead reduction and increase the scalability of the flooding approach with respect to traffic load. Therefore, one possible approach to solve our research question is to use a cluster-based control structure to increase the protocol scalability. A large MANET can be divided into several sub-populations or clusters by a clustering algorithm. However, a stable cluster-based control structure is harder to achieve where mobility is in presence. Therefore, there is a need to further investigate a clustering algorithm that can form a stable cluster-based control structure in a mobile environment.

2.4 Summary

A common method to allow higher protocol scalability is to subdivide network into smaller sub-groups. A clustering algorithm is usually used to divide the network into clusters. In this chapter, a detailed survey on different clustering algorithms proposed in literature for MANETs has been presented. Based on the literature survey, the research question is raised and solutions will be proposed in the following chapters.

On the other hand, multicasting is a more efficient technique to support group communication especially in resource-limited MANETs. Since the direct adaptation or extension of the wired/IP multicast routing protocols to MANETs appears unfeasible, much work has been done to design special multicast solutions for MANETs. In this chapter, a comprehensive survey has been conducted on the existing network-layer multicast solutions for MANETs. Despite the numerous protocols available, a satisfactory solution for MANETs is still not evident and there remain a number of issues and open problems that require further investigations and research, such as reliability, scalability, security, QoS support, and power consumption. In this research, robustness, efficiency and scalability issues of network-layer multicast protocols are highlighted.

Table 2.3 Comparison of tree-based multicast routing protocols for MANETs

	ABAM	MZR	DDM	MAODV	AMRoute	AMRIS	LAM	MDSR	ADMR
Initiative	Source	Source	Source	Receiver	Core (sender or member)	source (a special node)	Receiver/ source	Source	Source
Multicast forwarding Infrastructure	Tree (source-based)	Tree (source-based)	Tree (Source-based)	Tree (Shared-tree)	Tree (Shared-tree)	Tree (Shared-tree)	Tree (Shared-tree)	Tree (Source-based)	Tree
Underlying unicast	No	No	Yes (not specified)	Yes (AODV)	Yes (any)	No	Yes (TORA)	Yes (DSR)	No
Route discovery mechanism	On-demand	On-demand	On-demand	On-demand	Proactive	On-Demand	On-demand	On-demand	On-demand
Periodic overhead	No	Yes	No	Yes	Yes	Yes	No	No	No
Control overhead	Broadcast discovery packet. Tree formation (Route Reply & Setup). Tree repair (local / global)	Periodic advertisement (zone-restricted) Tree creation and repair (network-wide). Periodic tree refresh packet	Destination list in data packet header.	Tree formation and route discovery. Periodic hello and Group Hello message.	Periodic Join_Req (entire network – Mesh creation) Periodic Tree_create (within mesh – tree maintenance)	Tree formation and maintenance Periodic beaconing.	Tree joining and rejoining process.	Tree formation during DSR route discovery. Tree maintenance	ADMR header in all data packets. Keep-alive packets.
Route optimality	Good (Long-lived)	Moderate (Shortest path)	Moderate (Shortest Path)	Poor -suboptimal	Poor - suboptimal	Poor - suboptimal	Poor – suboptimal	Moderate (Shortest Path)	Moderate (Shortest delay path)
Scalability (# of senders)	Poor	Poor	Poor	Moderate	Moderate	Moderate	Moderate	Moderate	Poor
Loop-Free	Yes	Yes	Yes (if unicast is loop-free)	Yes	No	Yes	Yes	Yes	Yes
Node Leaving approach	Explicit leave message	Explicit pruning message	Explicit and soft state (timeout)	Explicit MACT (prune)	Explicit JOIN_NAK	Explicit SESSION-LEAVE message	Explicit LEAVE message	Route Error packet	No (passive ack)

Table 2.4 Comparison of flooding, mesh-based, hybrid and adaptive multicast routing protocols for MANETs

	Flooding	FGMP	ODMRP	CAMP	SMMRP	NSMP	DCMP	MCEDAR	M-LANMAR
Initiative	Source	Source/ receiver	Source/ Receiver	Receiver	Core Source	Source	Source	Source /Receiver	Receiver (Team)
Multicast forwarding Infrastructure	Mesh	Mesh (Source-based)	Mesh (Source-based)	Mesh (Core-based)	Mesh (Core-based)	Mesh (Source-based)	Mesh (Core-based)	Tree & Mesh	Tree & Flooding (Limited scope)
Underlying unicast	No	Yes (any)	No	Yes	No	No	No	No	Yes (LANMAR)
Route discovery mechanism	On-demand	Proactive / On-demand	On-demand	Proactive	Proactive	On-demand	On-demand	Proactive	Proactive
Periodic overhead	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes (Team maintenance)
Control overhead	No	Periodic flooding of membership advertisement. Periodic Forwarding/ Joining table	Periodic flooding of Join Req. Periodic flooding of Join Tables.	Mesh formation (Join Req + ACK) Mesh maintenance (heartbeat / Push Join)	Periodic flooding during mesh setup & reconfiguration. Mesh recovery.	Periodic global route discovery packet. Periodic local route discovery packet. New member request.	Periodic flooding of Join Req. Control packet for classifying passive and active sources.	Beaconing. Mesh setup and mesh maintenance.	Periodic broadcast of Landmark information. Periodic broadcast team membership.
Route optimality	Moderate	Moderate (Shortest path)	Moderate (Shortest Path)	Good (Shortest Path)	Good (Shortest path)	Moderate (reused forwarding nodes)	Suboptimal (core-based trees are used)	Moderate (Shortest path)	Moderate
Scalability (# of senders)	Poor	Moderate (FGMP-RA)	Poor	Moderate	Moderate	Moderate	Moderate	Moderate	Moderate
Loop-Free	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Node Leaving approach	No	No (soft state)	No (soft state)	Quit notification	Leave message	No (soft state)	Not mentioned	Leave message	Leave Message

Table 2.5 Comparison of tree-based approach versus mesh-based approach

	Mesh-based	Tree-based
Resource requirements	Bandwidth, processing and storage requirement to build a mesh is greater.	Bandwidth requirement to initialize a tree is lower.
Robustness	Redundant routes available. More robust in mobile scenarios.	Single path provided. Data loss increases with the mobility.
Efficiency	Number of data packet duplicated increases with the thickness of meshes.	More efficient as the data packet is only duplicated at tree branch.
Control Overhead	Usually use periodic flooding of control packets for mesh maintenance.	Frequent tree reconfiguration incurs substantial control overhead.

Table 2.6 Comparison of location-based multicast protocols for MANETs

	LBM	GeoTORA	GeoGRID
Initiative	Source	Source	Source
Multicast forwarding Infrastructure	Forwarding Zone	Forwarding Zone	Grid
Underlying unicast	No	Yes (TORA)	No
Route discovery mechanism	On-demand	On-demand	On-demand
Periodic overhead	No	No	No
Control overhead	Zone Information	Zone Information	Region Information Gateway Information
Route optimality	Poor	Moderate	Poor
Scalability (# of senders)	Poor	Poor	Poor
Loop-Free	Yes	Yes	Yes
Node Leaving approach	Not Applicable	Not Applicable	Not Applicable

CHAPTER 3

MOBILITY-BASED D-HOP CLUSTERING ALGORITHM

3.1 Introduction

In this chapter, a mobility-based *d*-hop clustering algorithm (MobDHop) [36] that forms *d*-hop clusters based on new mobility metrics is proposed. Inspired by Basu et al [52], two new mobility metrics, i.e. variation in ED between nodes over time (*VD*) and local variability value (*Var*) is proposed to be used in clusterhead election process. Since the mobility metric is used as clusterhead election criteria, the formation of clusters is determined by the mobility pattern of nodes. Mobile hosts in a MANET usually move in groups due to the nature of envisioned applications in MANETs. This is known as group mobility [126]. Due to team collaborations or group-based activities, mobile hosts usually have a common mission like saving victims that are trapped in collapsed building, perform a similar task like gathering information of threats in a battlefield or move in the same direction (rescue team designated to move towards east side of disaster struck area). Therefore, our algorithm attempts to capture the group mobility pattern and uses this information to form and maintain more stable clusters and thus provide a more stable underlying logical hierarchy to support other network control functions, such as network-layer multicasting. Section 3.2 presents assumptions used in MobDHop clustering algorithm.

Preliminary concepts and definition used in MobDHop will be presented in section 3.3. Section 3.4 contains a detailed description on the operation of MobDHop clustering include the cluster setup and maintenance phase. The correctness of MobDHop algorithm is also proved in this section. A brief summary is presented in section 3.5.

3.2 Assumptions

A successful dynamic clustering algorithm should achieve high cluster stability by forming and maintaining a stable cluster topology. These should be accomplished without prohibitive communications overhead and high computational complexity. Meanwhile, the efficiency of the algorithm is measured by the number of clusters formed. Therefore, the main design goals of our clustering algorithm are as follows:

i. The algorithm should minimize the number of clusters formed by considering group mobility pattern to achieve both efficiency and stability.
ii. The algorithm must be distributed and executed asynchronously.
iii. The algorithm must incur minimal clustering overhead, be it cluster formation or maintenance overhead.
iv. Network-wide flooding must be avoided.
v. Optimal clustering may not be achieved, but the algorithm must be able to form stable clusters should any exists.

The following assumptions are made:

i. Two nodes are connected by bi-directional link (symmetric transmission).
ii. The network is not partitioned.
iii. Each node can measure its received signal strength.

For the first assumption that all links must be bi-directional, this is to ensure all clusterhead can hear its member messages and vice-versa. This assumption can be relaxed if a neighbour discovery protocol that can identify uni-directional link is in presence. In this case, clustering algorithm can implement a special routine to handle uni-directional links. However, this is beyond the scope of this research.

Through periodic beaconing or hello messages used in some routing protocols, a mobile node can measure and record the received signal strength from that particular neighbour. In the Friss transmission equation, the received power over a point-to-point radio link is given by:

$$P_r = P_t * G_t * G_r * \frac{\lambda^2}{(4 * \pi * d)^2}$$

where P_r = received power, P_t = transmitted power, G_t = antenna gain of the transmitter, G_r = antenna gain of the receiver, λ = wavelength (c/f), and d = distance.

From the series of signal strength variations, statistical testing is applied to predict the relative mobility pattern between two nodes. Intuitively, two nodes are stably-connected if the received signal strength between them varies negligibly over time. If two nodes are moving together at a similar speed towards the same direction and their link is stable, the variation of their received signal strength should be very small. This serves as one of the metrics used in MobDHop to group the nodes into their respective clusters.

Complex calculation as proposed in previous works [127] to estimate physical distance between two devices is not needed in this thesis. In real world, it may not be possible to obtain an exact calculation of the physical distance between two nodes from the measured signal strength alone. It is important to note that MobDHop does not assume or require accurate estimations of physical distance between two nodes. Instead, a simple formula, which will be shown in the next section, is used to infer an “Estimated distance” (ED) between two nodes. This is not to estimate physical distance but to simplify the representation of signal strength and the “closeness" of two nodes as indicated by the received signal strength measured at the arrival of every packet from neighbouring nodes. The stronger the received signal strength, the “closer” the neighbouring node.

Thus, the smaller the ED is. It is important to know that the "closeness" between two nodes is not necessarily measured by their absolute or physical distance. For example, node *A* may be very close to node *B*. However, it is low in energy and transmits packets at lower power. In this case, it behaves like a distanced node from node *A*. Therefore, absolute distance is not useful in predicting link stability in this case.

Measured signal strength of successive packets is used to estimate the relative mobility between two nodes. The difference between EDs from a neighbouring node at two successive time moments is calculated. This difference indicates the pair-wise relative mobility as shown in Figure 3.1. If the new ED is larger than the old ED, the neighbouring node is moving away from the measuring node. Nodes are grouped into one-hop clusters based on their relative mobility in the first stage. Next, these one-hop clusters are expanded by merging individual nodes into one-hop clusters based on the previously described metric, i.e. the variation of ED between nodes in action. Before introducing MobDHop, a brief introduction on different terms and definitions is presented in the following section.

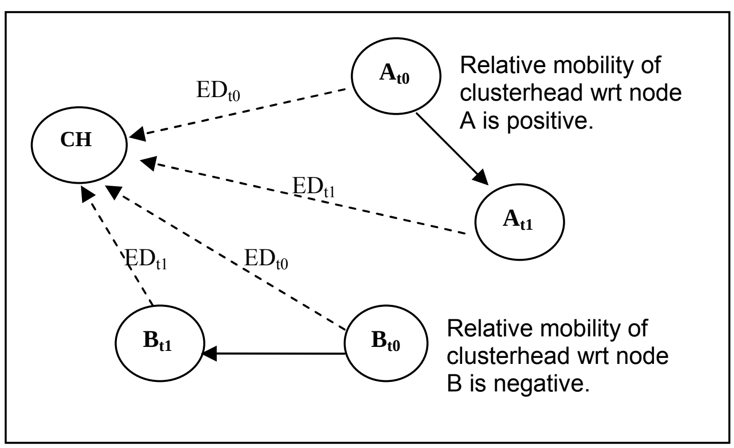


Figure 3.1 Illustration of relative mobility.

3.3 Preliminary Concepts and Definitions

A node may become a ***clusterhead*** if it is found to be the most stable node within its neighbourhood. Otherwise, it is an ***ordinary member*** of at most one cluster. When all nodes first enter the network, they are ***non-clustered***. A node may also become ***non-clustered*** when it loses its clusterhead due to node mobility. A node that is able to hear transmissions from another node in a different cluster is known as a ***gateway***. We formally define the following terms: (1) ED between nodes, (2) relative mobility between nodes, (3) variation of ED between nodes, (4) local variability, and (5) group variability.

Definition 1: ED between node A and B, $E[D_{AB}]$, is calculated as below. Please note that this formula is not aimed to obtain accurate physical distance between two nodes. Instead, it is a method to simplify the representation of signal strength (k is a constant and $k = 1$ in this study). P_r can be used directly in place of $E[D_{AB}]$ without affecting the correct operation of the algorithm.

$$E[D_{AB}] = \frac{k}{\sqrt{P_r}} \qquad \text{(Eq. 3.1)}$$

Definition 2: Relative mobility between nodes A and B, M_{AB}^{rel}, indicates whether they are moving away from each other, moving closer to each other or maintain the same distance from each other. To calculate relative mobility, we compute the difference of the distance at time, t and the distance at time, t - 1. Relative mobility at node A with respect to node B at t is calculated as follows:

$$M_{AB}^{rel} = E[D_{AB}^{t}] - E[D_{AB}^{t-1}] \qquad \text{(Eq. 3.2)}$$

Definition 3: The variation of $E[D_{AB}]$ over a time period, T, VD_{AB}, is defined as the changes of EDs between node A and B over a predefined time period. Let's consider node A as the measuring node. Node A has a series of ED values from node B measured at certain time interval for n times,

$E[D_{AB}] = \{E[D_{AB}]_t,\ t = 0, 1, 2, \ldots, n\}$. Therefore, we calculate VD_{AB} as the standard deviation of ED variation as follows:

$$VD_{AB} = \sigma\big(\ |E[D_{AB}]_1 - E[D_{AB}]_0|, \ |E[D_{AB}]_2 - E[D_{AB}]_0|, \ldots, |E[D_{AB}]_n - E[D_{AB}]_0| \ \big) \qquad \text{(Eq. 3.3)}$$

Definition 4: Local variability at node A, Var_A, represents the degree of variation in distance at node A with respect to all its neighbours. Local variability is the mean of variation of ED values of all one-hop neighbours. Therefore, it is calculated as follows:

$$Var_A = \mu\left(VD_{AB_1}, VD_{AB_2}, \cdots, VD_{AB_m}\right) \qquad \text{(Eq. 3.4)}$$

Definition 5: Group variability, $GVar_c$ for cluster, c, indicates the overall variability in one-hop cluster formed by MobDHop in the first phase. It is the mean of local variability value of all one-hop members in the cluster. The $GVar_c$ is calculated as follows:

$$GVar_c = \mu\left(Var_A, Var_{N_1}, Var_{N_2}, \cdots, Var_{N_m}\right) \qquad \text{(Eq. 3.5)}$$

3.4 Algorithm Description

MobDHop, a distributed algorithm, dynamically forms stable clusters which can serve as underlying routing architecture. First, MobDHop forms non-overlapping one-hop clusters like most of the existing clustering algorithms. Next, a merging process will be initiated when a non-clustered node requests to join the neighbouring cluster. A node may become non-clustered when it is newly activated or it loses its clusterhead due to node mobility. The merging process will only be successful if the newly formed cluster can achieve a required level of stability. Most of the existing clustering algorithms form one-hop clusters. MobDHop is designed to form *variable*-hop clusters that are more flexible in cluster diameter. The diameter of clusters is adaptive to the mobility pattern of network nodes. MobDHop only requires each node to know its one-hop neighbourhood and the clustering decisions are made independently at each node. The maintenance of MobDHop follows the mechanism suggested by Basagni in [49], i.e. each node continuously senses the surrounding topology and reacts accordingly when topology changes is detected. The following section describes the operation of MobDHop, which consists of two main phases: (1) Cluster Setup, and (2) Cluster Maintenance, in greater details.

3.4.1 Cluster Setup

To set up multihop clusters, MobDHop first performs an initial discovery procedure to form one-hop clusters. After a network is grouped into a number of one-hop clusters, an on-demand merging phase begins where non-clustered nodes may request to join the neighboring clusters. MobDHop ensures that the resultant cluster is stable and its diameter is no larger than the predefined maximum hop count. Details of both discovery and merging phase are elaborated in following sections.

3.4.1.1 Discovery Phase

First, MobDHop forms non-overlapping one-hop clusters (each cluster member is at most one hop away from its clusterhead). This process involves the computation of three mobility metrics: (a) variation of "Estimated Distance" (ED) between nodes over time (*VD*), (b) local variability (*Var*), and (c) group variability (*GVar*). When the network is first initialized, all nodes periodically broadcast Hello messages. Each node measures the received signal strength of every received Hello message and calculates the ED with respect to each neighbour based on **Eq. 3.1**. After receiving a pre-specified number of Hello messages defined by Discovery Interval, each node computes the *VD* with respect to every neighbour using **Eq. 3.3**. Figure 3.2 shows how the variation of ED over time (VD_{AB}) is computed by node A with respect to node B. Based on this information, each node computes a local variability value, *Var*, i.e. mean of *VD* of all neighbours (**Eq. 3.4**), which implies how stable a node is with respect to all immediate neighbours. Figure 3.3 shows the computation of Var_A by a clusterhead, *A*.

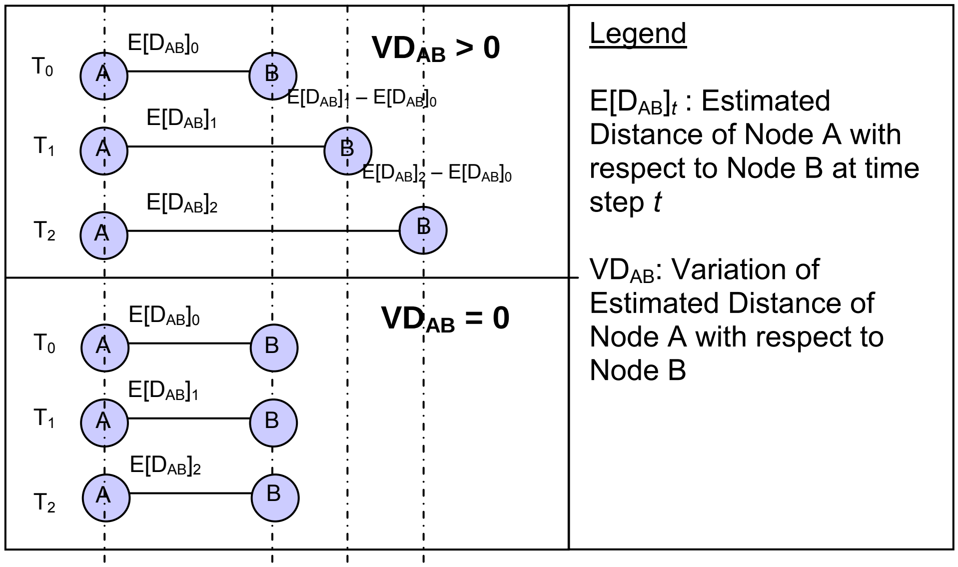


Figure 3.2 Computation of the variation of ED over time.

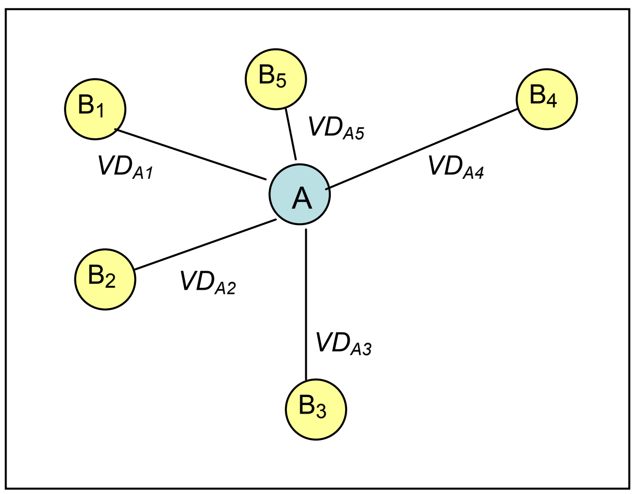


Figure 3.3 Computation of local variability value.

If a node has the lowest variability value, i.e. it is the most stable node among its neighbourhood; it assumes the role of clusterhead and announces it with a Hello message. Each clusterhead will compute group variability, *GVar*, i.e. mean of *Var* of its one-hop neighbours that join its cluster (**Eq. 3.5**). This value will be used in the merging phase in order to ensure a required level of stability can be maintained when allowing a new member to join the multihop clusters. Neighbour nodes assume the role of ordinary members. If a cluster member can hear Hello messages from more than one cluster, it assumes the role of a gateway. One-hop clusters will be formed by the end of the first phase regardless of mobility rate. Small clusters are formed in parts of the network which do not exhibit group movement. Small clusters in such areas are necessary due to the notion of spatial locality: "A mobile node cannot move too far too soon" [128]. Therefore, most topology changes are localized within a small area of the network for a certain period of time. By clustering nodes in these areas, local changes are abstracted and need not be seen by the entire network.

3.4.1.2 Merging Phase

After the discovery stage, all nodes are covered by one-hop clusters. A newly activated node or a node which is disconnected from its clusterhead due to mobility becomes non-clustered nodes. These non-clustered nodes will request to join the neighbouring clusters and this is the merging phase. The merging node will first observe its neighbourhood and choose the neighbour to which it is most stably connected. Then, it will try to merge into its neighbour's cluster if the following conditions are met:

- Hop count from a merging node to its new clusterhead is less than the parameter, d; if no restriction has been set (i.e. d = infinity), then this condition is irrelevant.
- The *VD* between the merging node and its chosen neighbour should be lower than the group variability (*GVar*) value computed by the relevant clusterhead multiplied by a factor, α. This factor, α, is introduced to control the stability level of the multihop clusters. A smaller α implies a stricter merging criteria and thus higher cluster stability can be achieved.

If a diameter restriction is imposed (d is set to a certain value), the first criterion ensures that the newly formed cluster will not grow beyond the maximum diameter, $2d$. The second condition ensures that the newly formed cluster achieves a required level of stability by taking their *VD* and cluster's *GVar* into consideration as shown in Figure 3.4.

After the merging process, a valid cluster structure will be achieved. Such a validity condition is defined by the following properties based on three ad hoc properties suggested by Basagni in [49]:

i. Every ordinary or gateway node has at least one clusterhead as its *d*-hop neighbour (*dominance* property).
ii. Every ordinary or gateway node affiliates with a clusterhead.
iii. No two clusterheads can be neighbours (*independence* property).

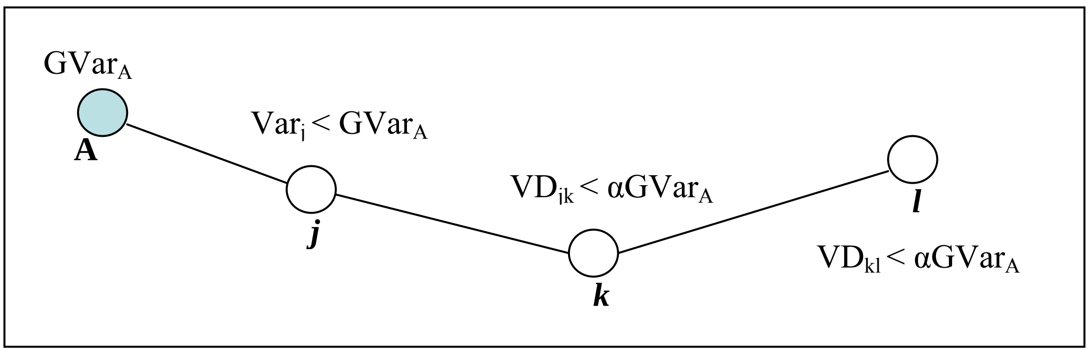


Figure 3.4 Merging process criteria.

3.4.2 Cluster Maintenance

Firstly, two cases that may cause topology changes in MANETs and thus invoke cluster maintenance are considered, i.e.:

i. A node switches on and joins the network.
ii. A node switches off and leaves the network.

When a node switches on, it is in the state of non-clustered. When a node switches off and this node is a parent node, this will cause its children nodes failing to receive cluster advertisements for a predefined period. These cluster members will also switch to non-clustered state. When a node finds itself in a non-clustered state, it will initiate merging process as described in Section 3.4.1.2 with neighbouring clusters whenever possible. If merging is not possible, it will declare itself to be a clusterhead of a one-node cluster. From time to time, it will try to merge with other clusters if possible.

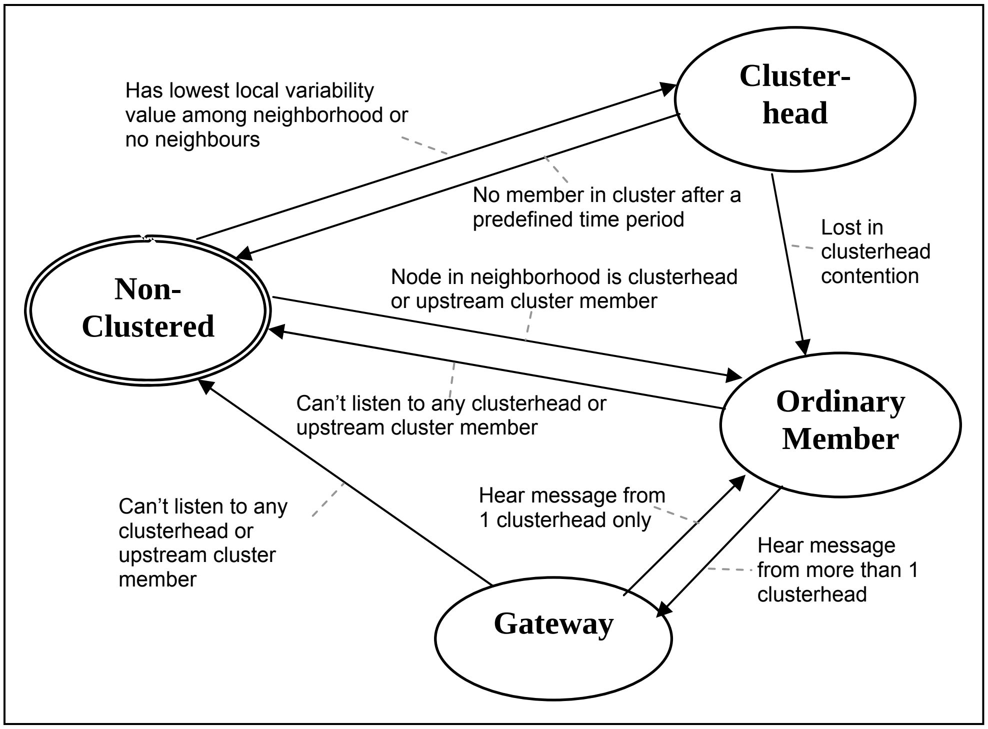


Figure 3.5 MobDHop node state transition diagram.

For topology changes that resulted from node mobility, cluster topology will be updated at each hello interval. Upon receiving hello messages from its neighbours, each node will update its neighbour table and react accordingly. Therefore, the node state transition is as shown in Figure 3.5. MobDHop is initiated by each node and continues to run for the entire lifetime of the node. As a member node moves around, it decides which cluster it currently belongs to and what role it currently plays based solely on the local information. Each node reacts to the changes in the surrounding topology and changes its status or cluster membership accordingly. When the link of an ordinary node to its parent node fails, the ordinary node will first try to merge into neighbouring clusters after ensuring that the stability can be preserved. If this fails, it will determine its new role in the same way as it does during initialization phase. Its children nodes will be notified of the clusterhead loss and react similarly. A stable and valid cluster structure can be re-established after a certain convergence period. If a node finds itself in a non-clustered state, it will attempt to merge with neighbouring clusters. Otherwise, it will declare itself to be a clusterhead of a one-node cluster, and periodically tries to merge with a neighbouring cluster.

3.4.3 Proof of Correctness

In this section, procedures for cluster set-up and cluster maintenance in MobDHop are presented in details. Apart from the abovementioned assumptions, we assume that a node *v*, knows its own ID *id*(*v*), its own local variability value *Var*(*v*), its own role *Status*(*v*) and the number of hop counts away from it clusterhead *H*(*v*). It is also aware of the ID, the local variability value, the role and the number of hop counts from clusterhead of all its one-hop neighbours. This can be done by implementation of a simple neighbour discovery procedure e.g. Hello Protocol. We also assume that local variability value which is used as primary clusterhead election criterion is unique (To relax this assumption, a secondary criterion can be used e.g. unique node ID.) The following four procedures are executed accordingly at each node *v* (Figure 3.6).

- ***Role_Assignment***. After a sampling period or discovery period where node *v* collects mobility information in its neighbourhood, node *v* executes procedure Role_Assignment to determine its role/status. If there is at least a non-clustered neighbour that has the lowest local variability value among all non-clustered neighbours, then node *v* will elect this node as its clusterhead. Otherwise, node *v* will become clusterhead if it has the lowest local variability value among all non-clustered neighbours.
- ***Update_Status***. Node *v* will execute this procedure periodically to keep itself updated with the neighbourhood information. Any link failure or link establishment will be made aware to node *v*. Then, node *v* reacts aptly against the changes in surrounding topology according to the following algorithm. If node *v* is clusterhead and it is made aware that another clusterhead has come into its transmission range, both nodes will execute *Clusterhead_Contention* procedure

to resolve this conflict. Clusterhead with lower local variability value will win the contention. If all cluster members leave its cluster, node *v* will try to run *Role_Assignment* procedure in order to reform a new cluster with its non-clustered neighbours. If node *v* is ordinary node, it may update its role to gateway node if it can hear hello messages from more than one clusterhead. If it hears no hello messages from its clusterhead, it will assume that it lost its clusterhead and execute *Merge* procedure to join neighbour's cluster which fulfils the stability and distance requirements. If node *v* is gateway node, it may update its role to ordinary node if it can only hear hello messages from its own clusterhead. If it hears no hello messages from its own clusterhead, it will assume loss of clusterhead and execute *Merge* procedure to attempt to join neighbour's cluster also.

- ***Clusterhead_Contention***. Node *v* executes this procedure when it is contending clusterhead role with another clusterhead when both of them are in each other transmission range. If node *v* has the lower local variability value, it will win the competition and announce itself as clusterhead. Otherwise it will give up its clusterhead role and make an announcement in the upcoming Hello messages. This will trigger a series of cluster changes since its cluster members will lose clusterhead and execute *Merge* procedure to join other clusters. We will look into the implication of clusterhead contention in terms of clustering overhead in the following chapter.
- ***Merge***. Node *v* executes this procedure to join a new cluster when it loses its clusterhead. Node v will first identify the neighbour node, *u* to which it has the lowest *VD* value (In other words, node *v* is most stably connected to node *u*.) Then, node *v* evaluates whether the following criteria are met:
    i. Hop count constraint is not violated.
    ii. Variation distance with respect to the target node does not violate the stability constraint.

If these criteria are met, node *v* will join cluster of node *u*. Otherwise, node *v* will choose another node, *w* to which it has the second lowest *VD* value and do the similar evaluation. If there is no node that can fulfil the merging criteria, node *v* will become a clusterhead.

To prove the correctness of the MobDHop algorithm, we have to show that MobDHop algorithm forms clusters and maintains clusters such that the cluster architecture is valid. The validity of cluster architecture is guaranteed by the following three properties:

i. Every node is eventually associated with a clusterhead to achieve valid cluster architecture.

ii. Every node is at most *d* hops away from its clusterhead.

iii. No two clusterheads can be neighbours.

Lemma 1: Every node is eventually associated with a clusterhead.

Proof: Each node enters the network and starts by executing *Role_Assignment* procedure. At the end of this procedure, each node will have either clusterhead role or ordinary role. The role and clusterhead of each node will be broadcast to its one-hop neighbours (tagged in the next upcoming hello message). If every ordinary node is associated to clusterhead node, then the cluster structure is valid. If an ordinary node *v*, chooses another ordinary node, *u* as its clusterhead, node *v* will run *Merge* procedure to identify new cluster to join. By the end of *Merge* procedure, node *v* may join a new cluster or become clusterhead itself. Hence, node *v* will be associated with a clusterhead. In one-hop Lowest-ID clustering, the worst case occurs when node ids are monotonically increasing or decreasing in a straight line as shown in Figure 3.7. A similar worst case scenario may happen in MobDHop algorithm when the local variability values of neighbouring nodes are monotonically increasing in a straight line. Hence, worst case convergence time for MobDHop algorithm is $\Theta(|V|)$. However, this configuration is highly unlikely in a real world scenario. A possible solution for this worst case scenario is to add a small random factor to the interval of Hello broadcasts. Then, nodes may not execute *Merge* procedure in a monotonically increasing fashion.

```
PROC Role_Assignment(id, Var)
id: the set of ID's of node v's one hop non-clustered
neighbours
Var: the set of local variability value of node v's one
hop neighbours
{
    if (id is empty)
    {
       Ch(v) = id(v);
       Status(v) = CLUSTERHEAD;
    }
   else if (id is not empty)
    {
             if (Var(v) = min(St))
             {
                Ch(v) = id(v);
                Status(v) = CLUSTERHEAD;
                H(v) = 0;
             }
             else
             {
                Ch(v) = id(min(Var));
                Status(v) = ORDINARY;
                H(v) = 1;
             }
    }
    broadcast_hello(id(v), Ch(v), Status(v));
}
```

```
PROC Update_Status(id, N)
N: the set of node v's one-hop neighbours.
id: the set of ID's of node v's one hop non-clustered
neighbours.
{
    if (Ch(v) = id(v) and Status(v) = CLUSTERHEAD)
    {
      if (Clusterhead Contention)
                Exec Clusterhead_Contention;
      if (Lost all cluster members)
                Exec Role_Assignment;
    }

    if (Ch(v) ∉ N(v) and
      (Status(v) = ORDINARY or GATEWAY))
                Exec Merge;

    if (Status(v) = ORDINARY)
    {
      if (hear messages from more
         than one clusterhead)
                Status(v) = GATEWAY;
    }

    if (Status(v) = GATEWAY)
    {
      if (hear messages from own clusterhead only)
                Status(v) = ORDINARY;
    }
}
```

```
PROC Clusterhead_Contention(id(u), Var(u))
id(u): id of node u, clusterhead in contention.
Var(u): local stability value of node u, clusterhead in
contention.
{
    if (Var(v) > Var(u))
    {
         Status(v) = ORDINARY;
         Ch(v) = id(u);
         H(v) = 1;
    }
    broadcast_hello(id(v), Ch(v), Status(v));
}
```

```
PROC Merge(VD, id, N, Var)
N: the set of node v's one-hop neighbours.
id: the set of ID's of node v's one-hop non-clustered
neighbours.
VD: the set of Variation of Distance (VD) value of
node v's one-hop neighbours
Var: the set of local variability value of node v's one
hop neighbours
{
    While ((VD is not empty) and
           (Status(v) = NONCLUSTERED))
    {
       u = min(VD);
       if ((H(u) + 1 < d) and (VD_uv ≤ Var(Ch(u))))
        {
                Ch(v) = Ch(u);
                Status(v) = ORDINARY;
                H(v) = H(u) + 1;
        }
        else
                VD = VD – {u};
    }

    if (Status(v) = NONCLUSTERED)
    {
        Ch(v) = id(v);
        Status(v) = CLUSTERHEAD;
        H(v) = 0;
    }
    broadcast_hello(id(v), Ch(v), Status(v));
}
```

Figure 3.6 Pseudocode for different procedures in MobDHop.

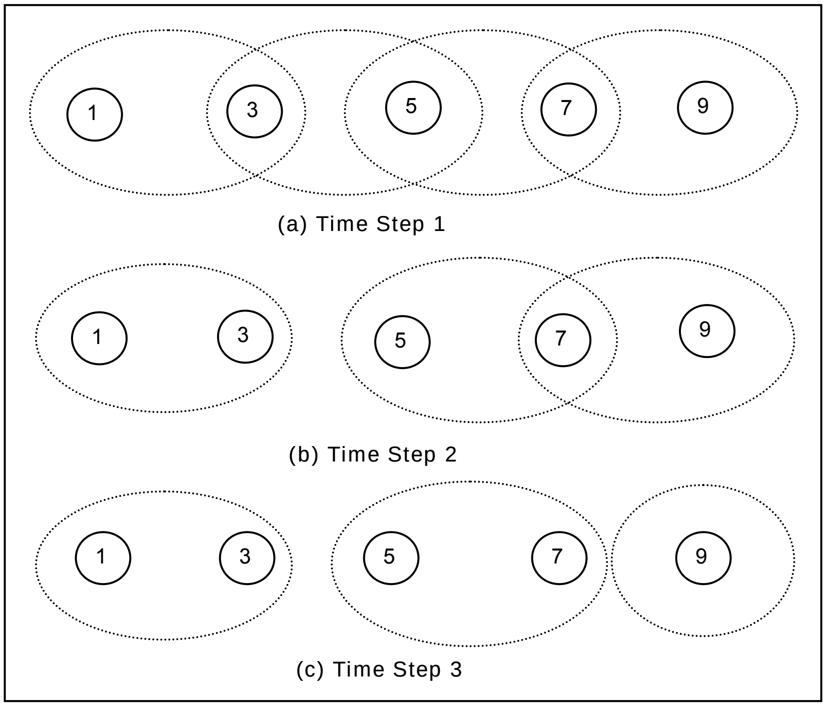


Figure 3.7 Worst case scenario with respect to convergence time of MobDHop.

3.5 Summary

In this chapter, Mobility-based *d*-Hop (MobDHop) clustering algorithm, a distributed algorithm that forms variable-diameter clusters that may change its diameter adaptively with respect to mobile nodes' moving patterns, is proposed. Three new mobility metrics i.e. Variation of ED over Time, Local Variability and Group Variability are proposed to form and maintain stable variable-hop clusters. The formation and maintenance of clusters in MobDHop are adaptive to the mobility patterns in the network to ensure maximum cluster stability. To achieve the desired scalability, MobDHop forms variable-diameter clusters, which allows cluster members to be more than one hop away from their clusterhead. The diameter of clusters is adaptive to the mobility behaviour of nodes in networks. Procedures for cluster setup and maintenance in MobDHop algorithm are presented and the algorithm correctness is also proved in this chapter.

CHAPTER 4

PERFORMANCE ANALYSIS OF MOBDHOP

4.1 Introduction

In this research, the performance of MobDHop clustering algorithm was examined via two approaches [37][38]: (1) simulation approach and (2) analytical approach. The performance of MobDHop was first evaluated via extensive simulations using NS-2 with CMU wireless extensions [34]. Simulation results demonstrated that MobDHop outperformed two well-know clustering algorithms, namely Lowest-ID clustering and Maximum Connectivity Clustering (MCC) by forming appropriate number of clusters with higher stability. The analysis of message and time complexity of MobDHop is also presented and it is shown that MobDHop, being a multihop clustering algorithm, only incurs O(1) control overhead per node per time step.

The use of MobDHop clustering algorithm to support unicast routing was also investigated in this research. Hierarchical approach has been used in literature, such as Clusterhead Gateway Switch Routing (CGSR) [45], Hierarchical State Routing (HSR) [129], Cluster-Based Routing Protocol (CBMP) [130], and Adaptive Routing using Clusters (ARC) [131], to improve the scalability of unicast routing protocol in MANETs. These protocols either form or assume clusters in MANETs that can serve as underlying control structure. Most of these protocols use ID-based clustering algorithm to form the cluster structure. To demonstrate how unicast routing can be supported by the stable two-tier cluster structure formed by MobDHop algorithm, we introduce a new variant of Ad hoc On-demand Distance Vector (AODV) protocol, namely MobDHop-AODV. MobDHop-AODV utilizes the topology aggregation knowledge available at each clusterhead to avoid unnecessary network-wide flooding of request packet in search for destinations. Simulations were conducted to evaluate the performance of MobDHop-AODV. Results and discussion were presented in section 4.6.

4.2 Evaluation Metrics

There is a lack of standardization in evaluation criteria of distributed clustering algorithms for MANETs. Deciding appropriate performance evaluation criteria for distributed clustering algorithm is non-trivial. Table 4.1 lists several criteria that should be taken into consideration during the evaluation of a distributed clustering algorithm.

To achieve protocol scalability by clustering, the number and the size of clusters formed should be optimized. Thus these criteria are critical in the evaluation. The clustering effort may not be useful if a large number of clusters are formed. Conversely, a clusterhead might not be able to handle all the traffic generated by its members if the cluster size is too large. A favourable clustering algorithm should therefore form appropriate number of clusters of moderate size.

In a highly dynamic MANET, the stability of the cluster structure is also a main concern. Instability of a cluster may affect the efficiency of the routing function and trigger additional clustering steps, thus incurring unnecessary overheads. A good clustering scheme should form clusters with stable cluster structure. In other words, the number of node transitions from one cluster to another should be minimized. Two cluster transitions/events involved that impact cluster instability are:

i. Election – an ordinary/gateway/non-clustered node becomes a clusterhead.

ii. Re-affiliation – an ordinary/gateway/clusterhead node leaves its cluster and joins another cluster.

To evaluate the cluster stability, the number of election and re-affiliation events, which occur at each time step is measured. The number of cluster changes, therefore, is the sum of these events. Cluster Residence Time (CRT) measures the average time that a mobile node stays with a cluster. Mean CRT is obtained by averaging the CRT of all nodes in the network. Hence, mean CRT represents the overall stability of a cluster structure. Normalized CRT can be used to compare cluster stability across different clustering algorithms, which is simulated under different scenarios. Coefficient of Variation (CoV) for CRT is also computed in our evaluation in order to provide a relative measure of data dispersion compared to mean CRT. The CoV is dimensionless and independent of scale. A high CoV indicates high variability of data. In our case, the deviation of CRTs compared to the mean CRT is small when the CoV is small. The combined consideration of these metrics fully reflects the stability of a cluster.

In most previous work, clusterhead lifetime or clusterhead duration is measured for each clusterhead and the mean value of clusterhead lifetime is used to represent cluster stability. Mean clusterhead lifetime is one of the stability measures used in related work [50][74][132][133]. Being a clusterhead for a very long period may over-drain limited resources of a mobile node [74]. Moreover, clusterhead lifetime may not be a fair metric to evaluate cluster stability. A node may play the role of clusterhead but its cluster may consist of a single node or its cluster membership may be highly dynamic. Therefore, these nodes may skew the mean value for clusterhead lifetime.

In most simulation evaluations, the assumption that continuous time is divided into discrete steps is made. Therefore, it is easier to measure the duration taken by an algorithm to establish or re-establish a valid cluster structure after a change in the network topology. This is called the convergence time or time complexity and is defined as the number of time steps from a topology change until a valid cluster structure is re-established. Convergence time and time complexity will be used interchangeably.

Meanwhile, the amount of knowledge required by each node in order to make clustering decision is another criterion. Each node must gather sufficient information before making any clustering decision. For example, each node in an ID-based clustering algorithm must know the identifier and role of all its one-hop neighbours in order to decide its own role while the Adaptive Clustering Algorithm [40] requires each node to know information of their one-hop and two-hop neighbours. Virtual backbone generation as proposed in [134] requires knowledge of its *r*-hop neighbours if the clusterhead is assumed to monitor nodes in its *r*-hop neighbourhood, where *r* is a predefined, fixed integer value.

The obvious drawback introduced by almost all clustering algorithms is the additional signaling overhead in order to maintain the cluster structure. Before implementing cluster architecture in a MANET, we must ensure that the benefits from clustering could outweigh the costs. Therefore, it is essential to investigate the amount of control overhead incurred by a clustering algorithm.

Table 4.1 Evaluation criteria for clustering algorithms for MANETs

Criteria	Description
Cluster Size	The number of members in the cluster.
Cluster Number	The number of clusters formed in the network.
Cluster Stability	The degree of stability a clustering algorithm can maintain in order to trigger as few cluster reorganization as possible.
Amount of Knowledge	The amount of knowledge a node needs to decide its cluster membership and its role.
Convergence Time	The number of time steps from a topology change until a valid cluster structure is established.
Clustering Overhead	The additional overhead incurred by a clustering algorithm in order to create and maintain a valid cluster structure.

4.3 Simulation Results of MobDHop

To evaluate the performance of MobDHop in MANETs, network simulations were conducted using NS-2 with CMU wireless extensions [34]. Simulation experiments were also conducted to compare the performance of MobDHop against two established clustering algorithms, namely Lowest-ID (L-ID) and Maximum Connectivity Clustering (MCC). These two clustering algorithms only require one-hop neighbourhood information like MobDHop whereas other multi-hop clustering algorithms such as Max-Min *d*-Clustering, (α,t)-Clustering, Connectivity-Based *k*-Hop Clustering require multiple-hop neighbourhood information which will cause excessive overhead in bandwidth-limited MANETs. Moreover, (α,t)-Clustering is specifically designed for Random Walk Mobility Model.

MobDHop, L-ID and MCC with LCC improvement were implemented in NS-2. LCC improvement ensures that clusterheads will only give up its role when: (1) another clusterhead comes into its communication range and wins the contention, or (2) it is disconnected from all its members. A similar maintenance algorithm was applied to all three clustering algorithms to justly compare their performance in both clustering setup and maintenance phase. In the following section, the simulation environment that was used is introduced. This is followed by an in-depth discussion on the simulation results obtained.

4.3.1 Simulation Environment

It was assumed that all nodes have identical and fixed radio transmission range, r (r = 125m). Two nodes were said to have a wireless link between them if they were within communication range of each other. Free space propagation channel model was used.

Two different mobility models, namely Random Waypoint (RW) [135] model and Reference Point Group Mobility (RPGM) model [126], were used in these simulations in order to emulate different motion behaviours of mobile nodes. In RW model, each node selected a random destination and moved towards it with a speed that was uniformly distributed between [*minspeed, maxspeed*]. Upon arrival, the node paused for a duration which is known as pause time and repeated the whole process until the end of simulation. In RPGM model, each group had a logical centre (group leader), which determined the group's motion behaviour. Initially, each member was uniformly distributed in the neighbourhood of the logical centre. Subsequently every node randomly moved with a certain speed and towards a certain direction with respect to the movement of its logical centre. However, each node may deviate from its group leader in speed, direction and distance, according to some predefined parameters. Different kinds of network scenarios were randomly generated with varying input parameters such as node number, node speed, maximum pause time etc. These scenario files were generated using BonnMotion, a mobility scenario generator and analysis tool developed by the University of Bonn [136]. The values of the various parameters used in the simulation are tabulated in Table 4.2, Table 4.3, and Table 4.4 respectively. Each simulation was executed for 900 seconds, similar to the simulation duration chosen by Basu et al [52]. It was observed that the statistics collected stabilized at 900 seconds and further execution of simulation does not lead to further variations in statistics collected. The state (role) of nodes is sampled at each second from 0^{th} second up to 900^{th} second. Each simulation was rerun for ten times with different seeds. Each data point was the average of ten series of data collected from simulation traces.

Table 4.2 Simulation parameters for all clustering algorithms

Configuration	Scenario 1	Scenario 2	Scenario 3
Mobility Model	RW & RPGM	RW & RPGM	RPGM
Number of Nodes	25,50,75,100	50,100,150	150
Network Area (m^2)	1000m x 1000m	1000m x 1000m	10 km x 10 km
Node Speed (m/s)	5,10,15,20,25,30	30	5 – 30
Max Pause Time (s)	30	0, 30, 60, 90, 120, 150, 180	30
Simulation Duration (s)	900	900	900
Purpose of Simulation	Varying Node Speed	Varying Pause Time	Varying Group Distance Deviation

Table 4.3 RPGM parameters

Parameter	Value in Our Simulation		
	Scenario 1	Scenario 2	Scenario 3
Group Size	5 (± 2)	5 (± 2)	15 (± 3)
Group Membership Change Probability	0.1	0.1	0.3
Max Group Distance Deviation (m)	10	5	50, 100, 150, 200, 250

Table 4.4 Algorithm parameters for MobDHop

Parameter	Meaning	Value in Our Simulation
BI	Broadcast Interval	0.75-1.25 sec
TD	Discovery Interval	BI * 6
TA	Assignment Interval	BI * 2
TM	Merge Interval	BI * 2
TC	Contention Period	BI * 2
MaxHop	Maximum Hop Count From Clusterhead	2 (if not specified explicitly)

4.3.2 Performance of MobDHop

The first series of simulations were to investigate the quality of a cluster structure formed by MobDHop under RW and RPGM model respectively by using network scenario 1 (cf: Table 4.2 and Table 4.3). The maximum hop count from the clusterhead was limited to two hops (cf: Table 4.4). Therefore, the diameter of each cluster might extend to at most four hops. Under scenario 1, node speed was varied from 5 m/s to 30 m/s to investigate the impact of node speed on the quality of the cluster structure formed by MobDHop. The quality of a cluster structure is reflected by its stability and its efficiency. The stability of a cluster structure was measured by mean CRT. A stable cluster structure should lead to a high value of mean CRT and CoV. Meanwhile, the efficiency of a cluster structure was measured by the average number of clusters formed per time tick and the average cluster size per time tick. An efficient cluster structure should not consist of a large number of small clusters.

Figure 4.1(a) and (b) show the impact of the average node speed on the stability and the efficiency of the cluster structure formed by MobDHop respectively under RW model. As shown in Figure 1(a), the stability of the cluster structure, which was indicated by the mean CRT, dropped significantly with the increase in the average node speed under RW model. A similar trend was observed in all different sizes of networks. Thus, cluster stability decreased with the increase of mobility rate regardless of the number of node in the network. This can be explained by the fact that cluster stability is inversely related to the number of cluster topology changes incurred in the network. A similar observation was made by other researchers in respective research on DMAC clustering [75], (*a*,*t*)-clustering [58] and Random Competition Clustering [133]. These cluster topology changes are usually attributed to the inevitable wireless link breaks. According to the analysis by Sucec and Marsic [137], the rate of wireless link break increases with the average node speed, network size and network density if all network nodes move in a random fashion according to the definition of RW model. Therefore, wireless link breaks are more often in a highly mobile random network (high average node speed) than a semi-static random network (low average node speed). Wireless link breaks are also more often in denser networks with similar network rate. The density of a network usually increases with the growth of the number of nodes in a constant-sized network if the transmission range remains unchanged. It was observed that network with 25 nodes achieved slightly higher stability than its counterparts. This could be attributed to the fact that the 25-nodes network was much sparser than network with more nodes (50, 75 and 100 nodes respectively) and therefore the rate of wireless link break was lower according to the analysis done by Sucec and Marsic [137]. Furthermore, a fewer number of clusters were formed in 25-node network as shown in Figure 4.1(b). It was found that the efficiency of the cluster structure, as indicated by the average number of clusters formed and the average cluster size, increased with the density of simulated networks as shown in Figure 4.1(b). For instance, the network with 100 nodes consisted of an average of 15 clusters with 4 nodes per cluster at every instance of time. The increasing cluster size is attributed to the fact that a clusterhead may have more neighbouring nodes in a denser network. These results agreed with the findings from [50] where the authors evaluated Max-Min *d*-Clustering, L-ID clustering and MCC.

The authors [50] observed that the average number of clusters formed and the average cluster size increased with network density.

On the other hand, Figure 4.2(a) and (b) show the impact of average node speed on the stability and the efficiency of the cluster structure formed by MobDHop under RPGM model. As shown in Figure 4.2(a), the stability metric, which was mean CRT, was not related to the average node speed as in Figure 4.1(a). These results suggest that MobDHop could identify group mobility pattern correctly and assign nodes which move in a similar pattern to the same cluster. Therefore the cluster structure formed by MobDHop was not affected by the mobility rate. However, it is observed that mean CRT decreased with node density. This can be attributed to two main reasons: (1) individual movements of nodes due to speed and direction deviation from group leader defined by RPGM model, and (2) more clusters are formed in network with higher density as shown in Figure 4.2(b). This implies that a larger number of nodes are elected as clusterheads in the network. Therefore, there is higher chance that two clusterheads will come into the transmission range of each other and the clusterhead contention will take place more frequently. Clusterhead contention will cause one of the clusterheads to give up its role and all cluster members will be involved in cluster re-affliation events. These re-affliation events reduce the stability of the cluster structure formed and lower the mean CRT observed.

By comparing Figure 4.1(a) and Figure 4.2(a), mean CRT under RPGM model was observed to be much higher (about ten times higher) than mean CRT under RW model. This implies that clustering may not be beneficial in RW model since the quality of the cluster structure was not a satisfactory. However most of the previous research [40][46][48][49][50][52][56][57][58] evaluated the performance of their clustering algorithm based on RW model. There are hardly any results available for RPGM model. On the other hand, Figure 4.2(b) shows that the clusters formed under RPGM model consisted of five members which conformed to the RPGM group size parameter used in the simulations as shown in Table 4.3. This suggests that

MobDHop could identify the group mobility pattern and form the appropriate number of clusters in a MANET efficiently.

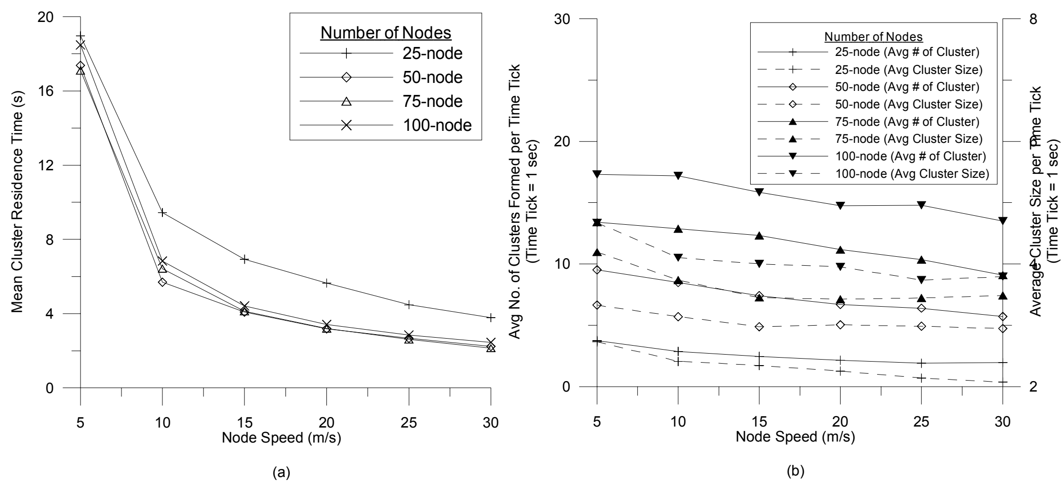


Figure 4.1 Impact of node speed on (a) cluster stability and (b) average number of clusters and average cluster size under Random Waypoint Model.

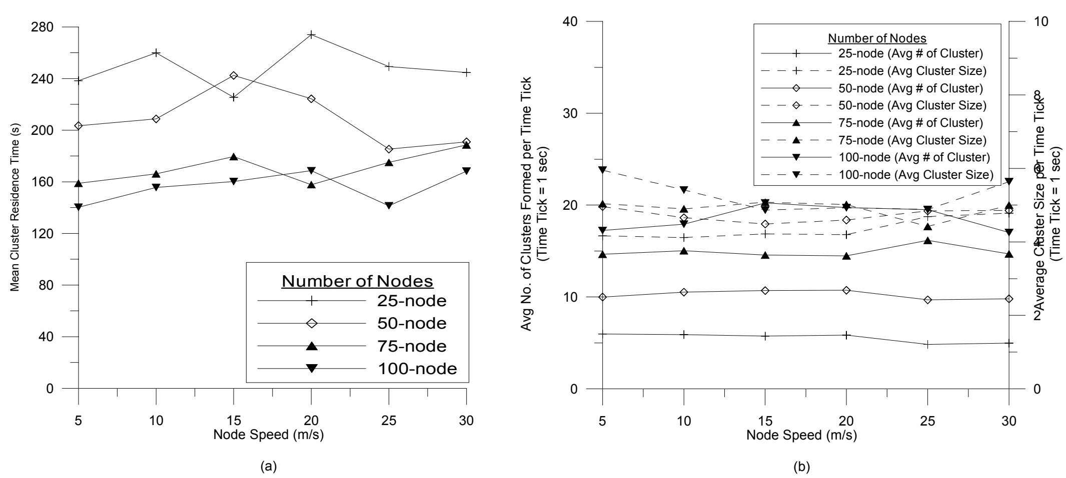


Figure 4.2 Impact of node speed on (a) cluster stability and (b) average number of clusters and average cluster size under Reference Point Group Model.

Figure 4.3 displays the impact of node pause time on the stability of the cluster structure formed by MobDHop under different mobility models. There are two main observations in these results. First, mean CRT under RW model increased with the increase of pause time while mean CRT under RPGM model was not influenced by the duration of pause time. Second, mean CRT under RPGM model was about 15 times higher than mean CRT under RW model. In a MANET under Random Waypoint mobility model, a longer pause time means the network is more stationary. Therefore, the network is subjected to less topology changes due to wireless link breaks. Under such circumstance, cluster structure formed by MobDHop may experience less cluster topology changes and less cluster re-organizations will be invoked. This explains the rise of mean CRT under RW model with the increase of the duration of pause time. However, this does not apply to a MANET under RPGM model. The results suggest that networks under RPGM model yield much longer mean CRT due to the fact that their group mobility pattern could be correctly captured by MobDHop. Hence, clusters were much more stable. However, the lower stability measure in denser networks is due to the similar causes as in network scenario 1. Nodes may enter the coverage area of other clusters at a higher probability in a denser network. If the node is a clusterhead, it causes clusterhead contention. Clusterhead contention triggers clusterhead re-election and cluster re-affiliation events that lead to shorter cluster residence durations.

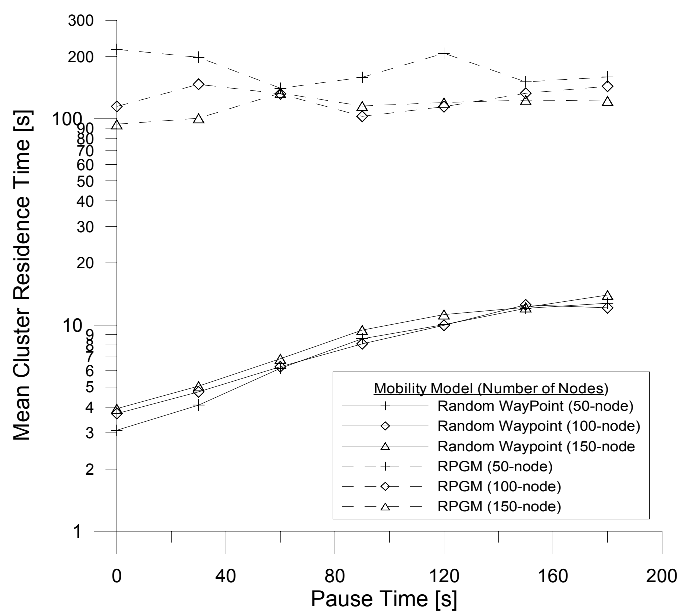


Figure 4.3 **Impact of pause time on cluster stability under different mobility patterns (y-axis is shown in $\log_{10}$ scale).**

4.3.3 Performance Comparison

Next, the performance of MobDHop was compared against existing clustering algorithms. First, network scenario 1 (RW model) was used to evaluate cluster stability and clustering efficiency of different clustering algorithms in relatively small MANETs which exhibited random waypoint mobility. The network density increased with the rise in the number of node. Figure 4.4, Figure 4.5, Figure 4.6, and Figure 4.7 show the impact of node speed on cluster stability for a 25-node, 50-node, 75-node and 100-node MANET respectively. MobDHop (1-hop) formed one-hop clusters similar to L-ID and MCC algorithm, in which all nodes are at most one hop away from their clusterhead. From the results, both variations of MobDHop outperformed L-ID algorithm in all scenarios. Both variations of MobDHop formed clusters that had longer CRT and lower CoV for mean CRT than their competitors. This shows that both variants of MobDHop formed clusters that were more stable. A lower CoV also indicates the scatter of CRTs obtained in MobDHop compared to the mean CRT was much smaller than those measured in L-ID and MCC clustering algorithms. They also initiated less cluster changes (election and re-affiliation events) than the L-ID algorithm. It is observed that 1-hop MobDHop performed slightly better than multihop MobDHop in small MANETs. This is because the simulated network does not exhibit any group mobility pattern under RW assumption. Therefore, only one-hop clusters should be formed to capture localized mobility. However, multihop MobDHop formed multihop clusters which are less stable. Forming these multihop clusters in random networks caused higher number of clusterhead re-election and re-affiliation events as cluster stability was harder to be maintained under random mobility. Lower CRT was therefore yielded. In MobDHop, both *GVar* and *d* are used to control the growing of cluster diameter. As the nodes move randomly, the group variability (*GVar*), which is the mean of local variability value of all one-hop members of the clusterhead, was much higher. Therefore, the cluster allowed nodes which are relatively instable to merge into the cluster since GVar which is used as control parameter has a larger value. Therefore, this leads to the formation of less stable multihop cluster. This problem can be alleviated by setting α to a smaller value and thus a stricter merging criterion can be imposed. Another alternative is to set suitable *d* for MobDHop algorithm according to the mobility pattern in the network.

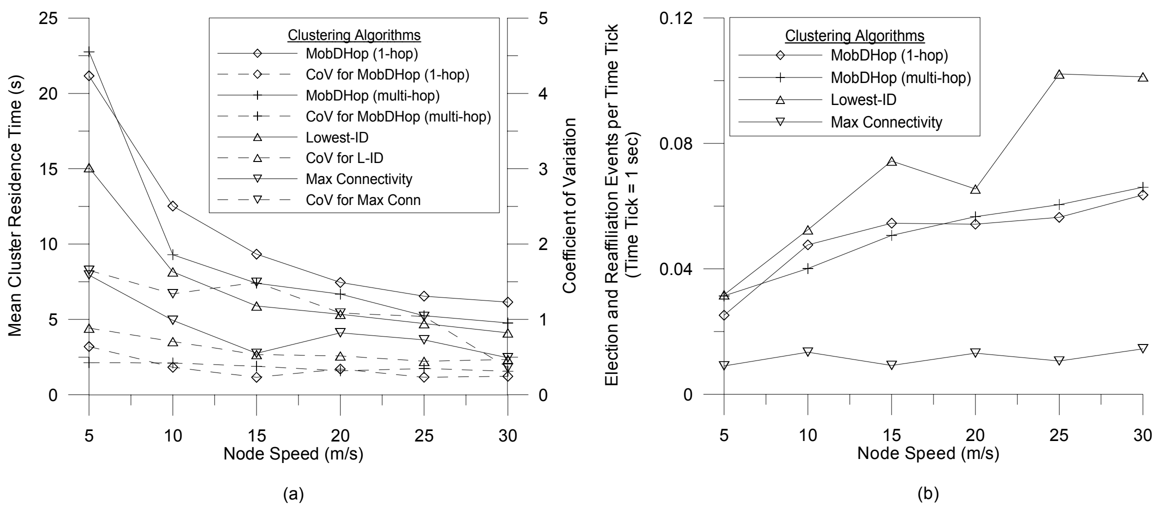


Figure 4.4 **Impact of node speed on cluster stability for a 25-node MANET under Random Waypoint Model.**

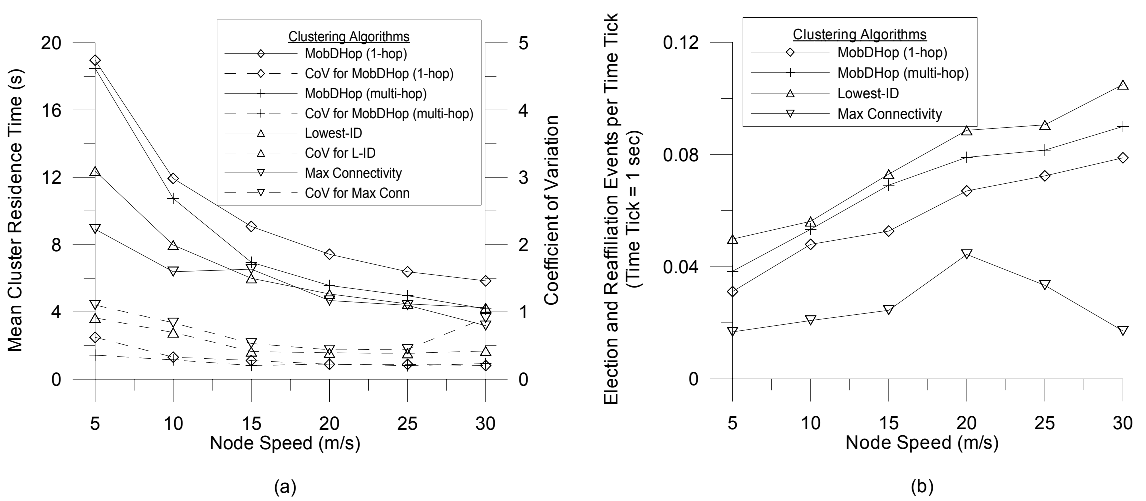


Figure 4.5 **Impact of node speed on cluster stability for a 50-node MANET under Random Waypoint Model.**

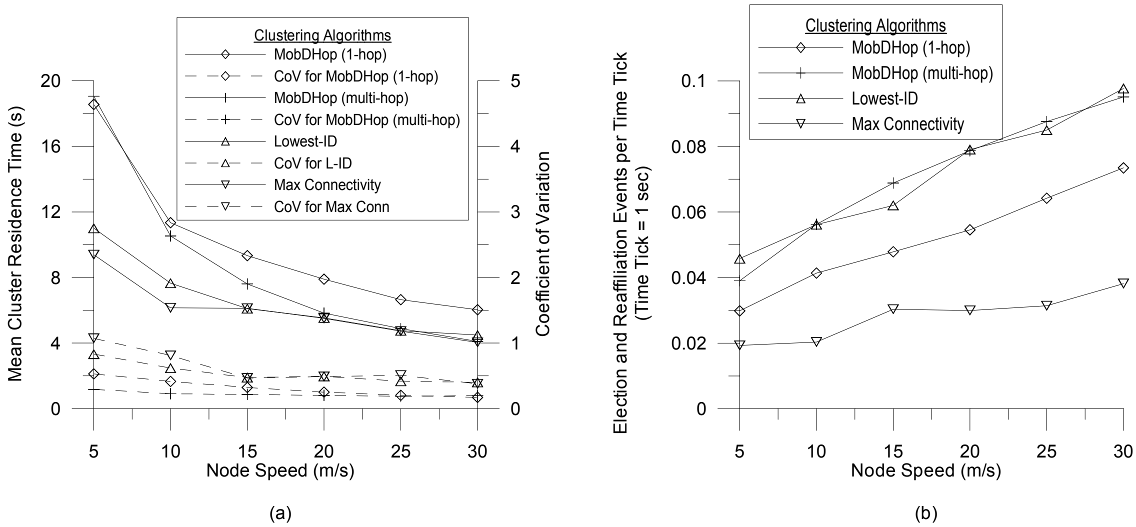


Figure 4.6 **Impact of node speed on cluster stability for a 75-node MANET under Random Waypoint Model.**

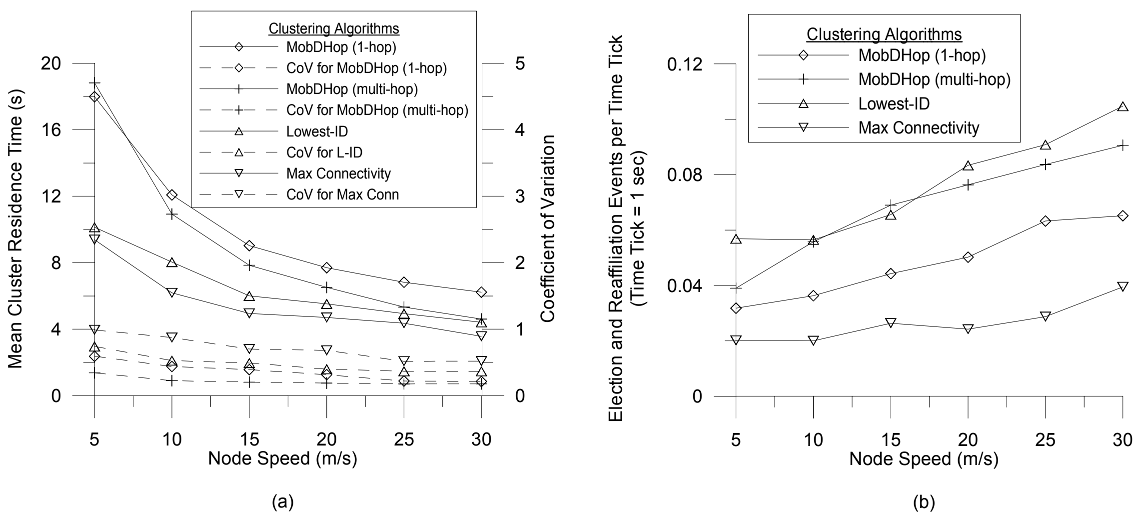


Figure 4.7 **Impact of node speed on cluster stability for a 100-node MANET under Random Waypoint Model.**

While the MCC algorithm formed the most instable clusters (shortest CRT), it has been observed that, counter-intuitively, MCC initiated the least cluster changes. It is important to note that this observation does not imply that MCC has the best performance among all algorithms. Figure 4.8 displays that MCC formed fewer clusters that were generally smaller in size. This implies that a large portion of nodes remained un-clustered or became single-node cluster. Due to the fact that MCC chooses clusterhead that has maximum number of neighbours during cluster setup, clusterhead lifetime is usually longer under MCC with the LCC improvement. Hence, clusterhead election seldom takes place. In the MCC algorithm, a node will only choose "uncovered" neighbouring nodes to be their clusterhead. When a node is disconnected from its clusterhead, it will first try to choose a clusterhead in its neighbourhood which has higher connectivity degree to be its clusterhead. If this could not be found, it will try to run the clusterhead election algorithm with its neighbouring "uncovered" nodes. With the LCC improvement, a clusterhead will not give up its role until all members have left its cluster. Thus, it is highly likely that both of the abovementioned conditions could not be met and a non-clustered node will declare itself as a clusterhead. As a result, a large number of single-node clusters are formed. Since there is no member in a single-node cluster, it is not possible for re-affiliation events to happen and this leads to a low number of cluster changes. Thus, it is noteworthy to mention that all performance metrics are related and the performance of an algorithm has to be carefully examined by taking all metrics into consideration.

Figure 4.8 shows the impact of node speed and network density on the number of clusters formed by these clustering algorithms. As expected, multihop MobDHop formed less clusters than its one-hop counterpart, L-ID and MCC in 25-node, 50-node, 75-node and 100-node MANETs. Multihop MobDHop also formed slightly larger clusters. In conclusion, MobDHop is favoured over L-ID and MCC algorithm since it forms less clusters and the clusters are less volatile.

The quality of cluster structure formed by MobDHop and other clustering algorithm in sparse and large MANETs was also evaluated and compared in another series of simulations. Network scenario 3 (cf: Table 4.2 and Table 4.3) features a very large MANET (10km x 10km), which consists of 150 nodes that are members of 10 different groups and these groups move in different direction and speed. The main purpose of these simulations was to verify the performance of various clustering algorithms under different kinds of group behaviours. Therefore, the group distance deviation parameter (a parameter that indicates the distance allowed for group members to deviate from the group leader) in RPGM model was varied accordingly to produce different group scenarios. Some realistic scenarios include (1) group members moving together and remaining close to each other (small group distance deviation), and (2) each member is in-charge of one small area but they still communicate with one another for information exchange (large group distance deviation). Besides, the maximum hop count parameter, d in MobDHop was varied to 1, 2, 3, 4 and *infinity* (denoted as v, the largest integer value in simulation).

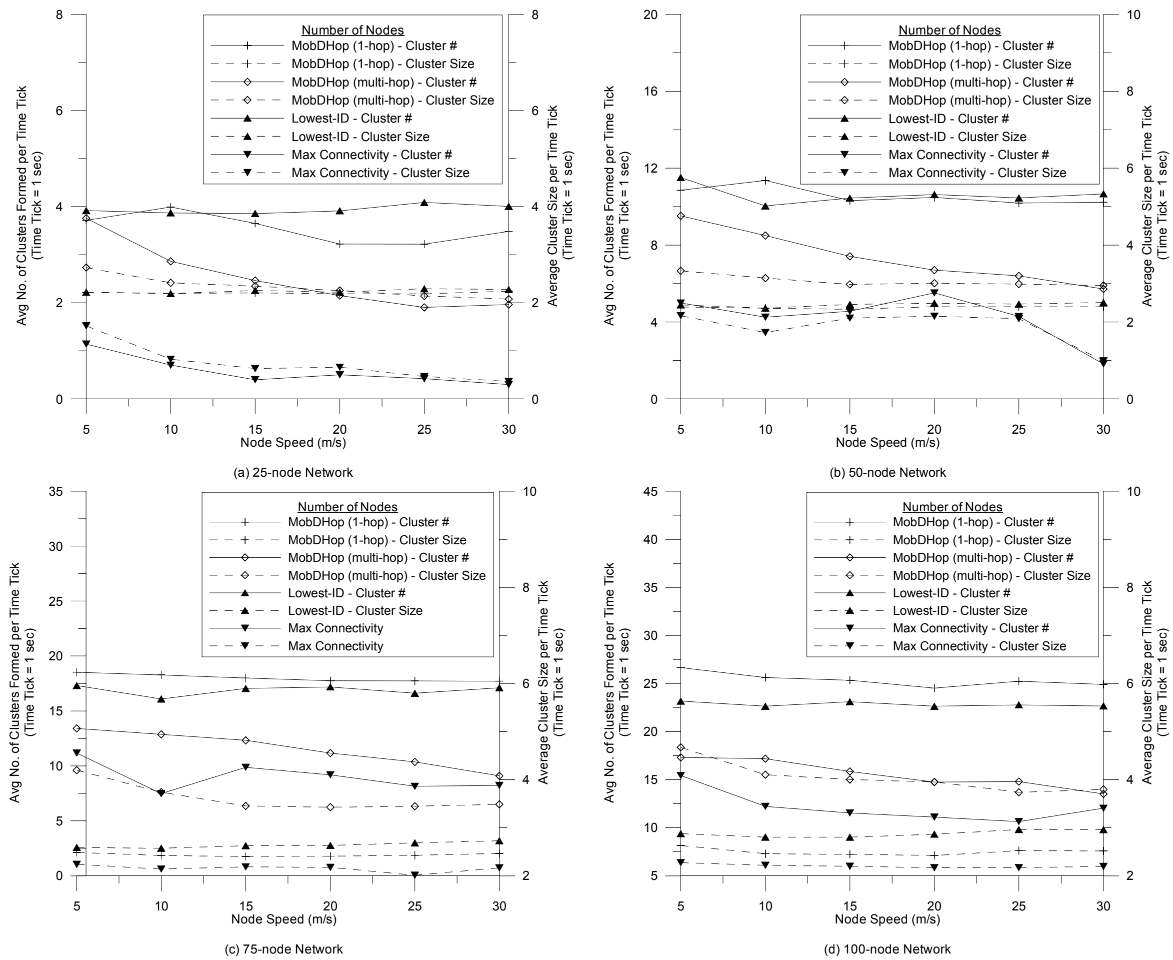


Figure 4.8 Impact of node speed and network density on the number of clusters formed for MANET under Random Waypoint Model.

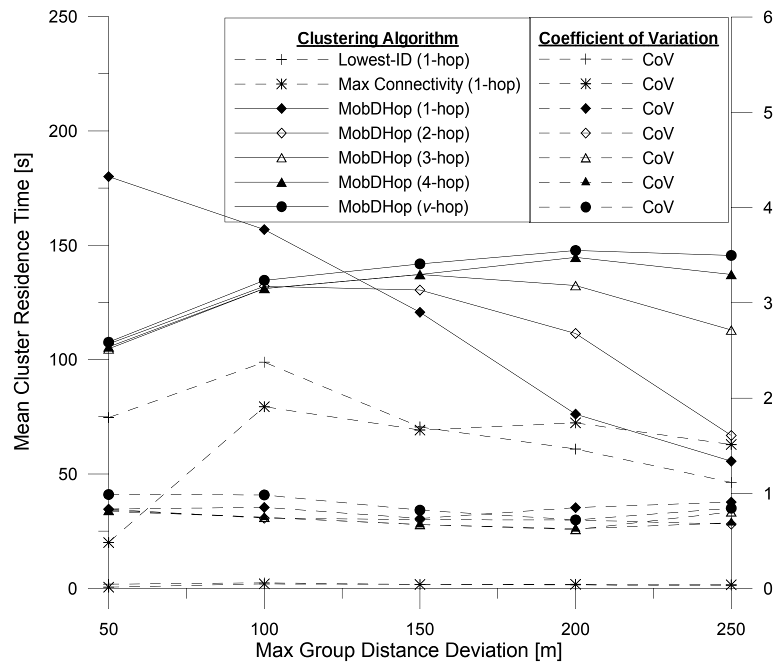


Figure 4.9 **Impact of maximum group distance deviation on CRT.**

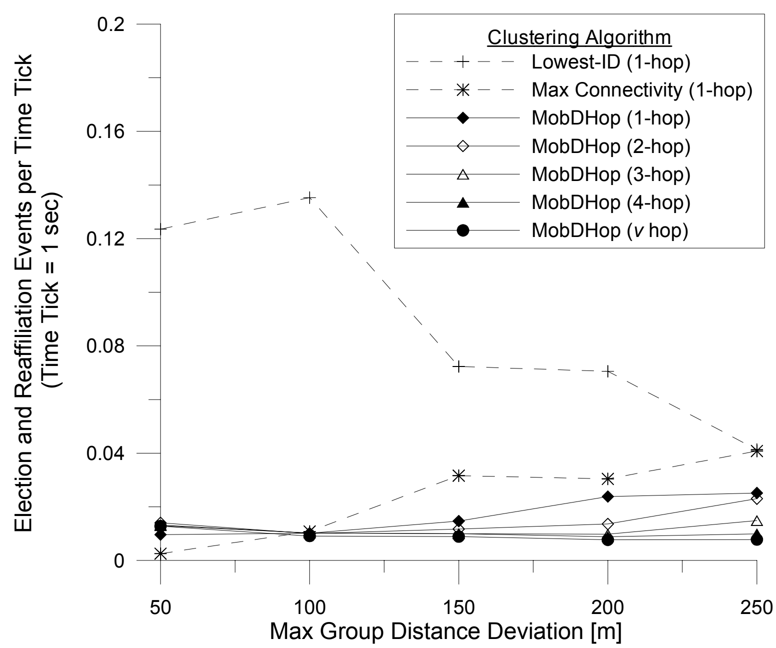


Figure 4.10 **Impact of maximum group distance deviation on election and re-affiliation events.**

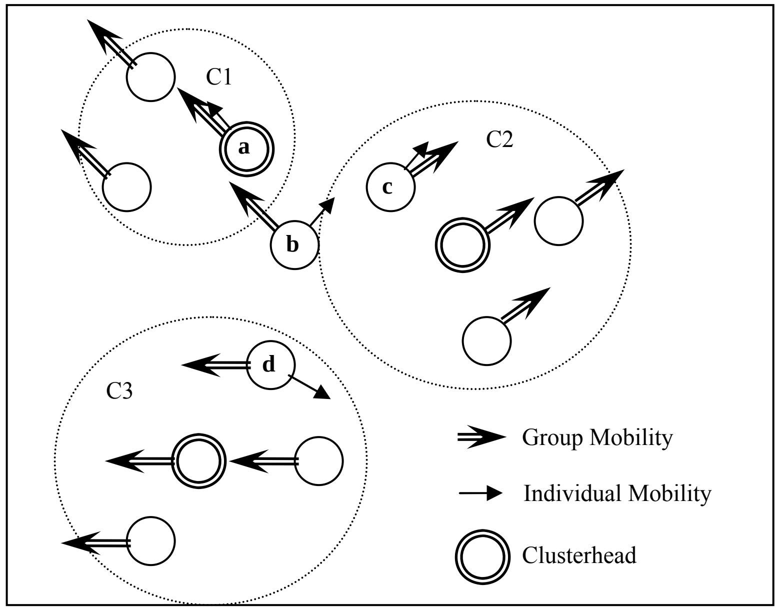


Figure 4.11 Mis-clustering event in multihop MobDHop.

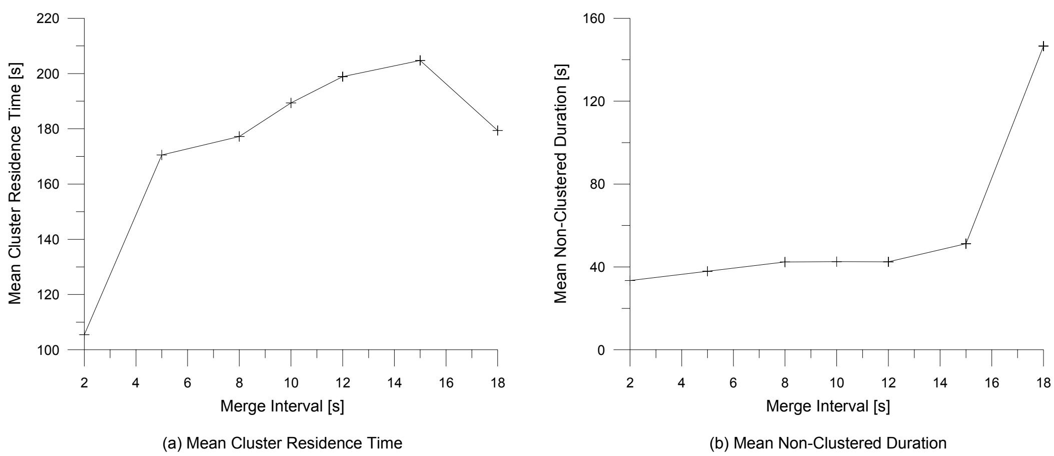


Figure 4.12 Impact of the duration of merge interval in RPGM network.

Figure 4.9 shows the impact of group distance deviation on the mean CRT and Coefficient of Variation (CoV) of clusters formed by different clustering algorithms. As shown in Figure 4.9, all variations of MobDHop algorithm formed cluster that were much more stable (longer mean CRT) than their competitors (L-ID and MCC). Figure 4.10 shows the number of election and re-affiliation events per second incurred by different clustering algorithms. Consistent with Figure

4.9, MobDHop algorithm outperformed L-ID and MCC algorithms by incurring lower number of clusterhead election and re-affliation events.

1-hop MobDHop formed the clusters with longest mean CRT when group distance deviation was less than transmission range, 125 metres. However, its performance dropped drastically with the increase of group distance deviation. In the contrary, all variations of multihop MobDHop outperformed their one-hop counterpart when the group deviation distance was larger than 125m. This may be due to the fact that most of the group members remain in the immediate neighbourhood of their clusterhead when the group distance deviation is less than group leaders' one-hop transmission range. Forming multihop clusters in this scenario may lead to the decline in cluster stability. Mis-clustering, which happens when the node chooses to join the cluster with different mobility behaviour, is more likely to happen when multihop clustering is allowed as shown in Figure 4.11. For instance, node *b* which wishes to join into neighbouring clusters will first observe the link condition with respect to all possible neighbours before making re-affliation decision. Since multihop clustering is allowed, there are more potential neighboring clusters that node *b* can merge into (Cluster C1, C2 and C3). Due to a relatively short merge interval, node *b* may observe a fairly stable link with one of the potential neighbours since two nodes from two different groups may exhibit temporary similar moving pattern due to their individual mobility as defined in RPGM model. In the following example, node *b* will merge into cluster C2 via the link with node *c* since both of them exhibit similar moving pattern for a short period of time. When the link finally breaks, cluster topology change takes place and cluster stability is therefore affected.

To verify the impact of the duration of merge interval (which is the time interval where a non-clustered node gathers and computes its *Var* with respect to all its one-hop neighbours) in the above-mentioned scenario, another set of simulations were executed by using *v*-Hop MobDHop and the duration of merge interval was varied from 2 seconds to 18 seconds in RPGM network with maximum group distance deviation of 50 metres and other parameters were similar to those in network scenario 3. The results as shown in Figure 4.12 indicate that the increase in the duration of merge interval could improve the cluster stability. This was reflected by the significant increase in mean CRT. However, it is also observed that the amount of time a node stayed un-clustered also increased with the lengthening of merge interval. This shows that the duration of merge interval has to be carefully chosen in order to form stable clusters without causing the node to remain un-clustered for long period.

Multihop MobDHop performed better by incurring less cluster changes and longer CRT as the group distance deviation increased. This might be explained by the fact that most of the group members are located out of group leaders' one-hop transmission range in these scenarios. Therefore, multihop MobDHop could form corresponding multihop clusters. Figure 4.13 also shows that multihop MobDHop formed least number of clusters with an average size of 10-15 members.

Another important observation was made when the hop count parameter *d* was set to a very large value (*v*-hop MobDHop). The results of *v*-hop MobDHop were almost indifferent from the results of 4-hop MobDHop. This shows that MobDHop adaptively forms stable clusters based on group moving patterns. This could not be achieved by other *k*-hop clustering algorithms as those algorithms require *k* to be predefined in order to determine cluster diameter. If *k* were to be set to a large value in those algorithms, all network nodes will be clustered into a single cluster if *k* is larger than the network diameter. To further verify this claim, we plotted the average maximum cluster radius per time tick against maximum group distance deviation in Figure 4.14 for the case where *d* was set to a very large value (*infinity*). When maximum group distance deviation was set to 50m, the average maximum cluster radius was 1. This implies that most of the clusters formed by MobDHop in this scenario consisted of one-hop clusters. When maximum group distance deviation was increased to 250m, the average cluster radius was about 3.5. Instead of forming one large cluster, MobDHop formed an appropriate number of clusters with average maximum radius of 3.5.

In short, multihop clustering is favourable when the group distance deviation becomes larger and group members are out of immediate transmission range of the group leader. In these scenarios, the group leader has to communicate with its member via multihop links. To better facilitate group communication which is a norm in collaborative applications, the members of same group should be clustered.

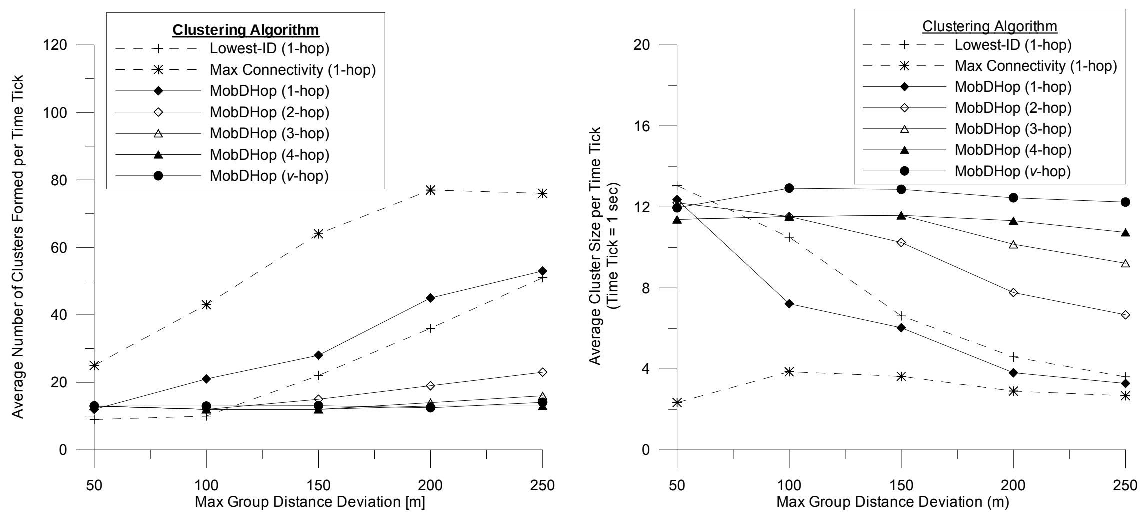


Figure 4.13 **Impact of group distance deviation on the average number of clusters and average cluster size per time tick.**

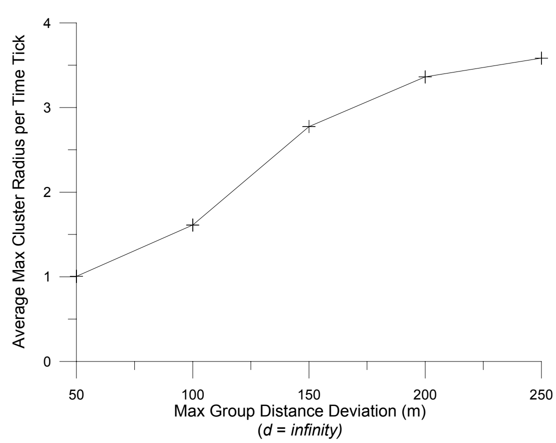


Figure 4.14 **Impact of group distance deviation on the average maximum radius per time tick under RPGM.**

4.4 Analysis of Time and Message Complexity

The goal of this section is to evaluate MobDHop clustering algorithm with respect to its clustering overhead and convergence time. The following section presents our in-depth analysis. As mentioned earlier, clustering overhead incurred by MobDHop clustering algorithm, ψ_C, consists of:

i) Hello Protocol Overhead (ψ_H)

ii) Cluster Formation Overhead (ψ_{CF})

iii) Cluster Maintenance Overhead (ψ_{CM})

Total clustering overhead per node incurred by MobDHop clustering algorithm is the sum of the above contributing factors. The following claim is made regarding the average clustering overhead per node per time step:

Claim 1: ψ_C = O(1) packet transmissions per node per time step.

4.4.1 Assumptions

A MANET is represented by a connected, undirected graph, $G = (V, E)$, where V is the set of nodes and E is the set of bidirectional links. We assume that nodes are located randomly throughout the network area and they are not initialized at the same time. (This is to eliminate the possibility of having the monotonically increasing or decreasing IDs as mentioned in Section 3.4.3). To simplify the analysis of link change frequency, the random waypoint mobility model with zero pause time is assumed. Two nodes are neighbours if their Euclidean distance between each other is less than their transmission radius R. We also assume that a message sent by a node is received correctly within a finite time (a time step) by all its neighbours.

4.4.2 Definitions

The following definitions will be used in the following analysis.

- N = the number of nodes in the network.
- m = the average number of members in a cluster, $0 \leq m \leq N$.
- D = the duration of communication session.
- r_{hello} = the number of hello messages emitted by a node per time step (hello rate).
- r_{link} = the average number of link state change events occurred per time step.
- μ = average node speed.
- h_i = hop count from clusterhead of node i.
- H_{max} = maximum hop count from clusterhead.
- T_{sample} = the number of time steps taken by a node to collect stability information from neighbours.
- T = the number of time steps taken by the algorithm after a change in the topology to accomplish cluster reorganization (Time Complexity).
- M = the number of messages exchanged between nodes after a change in the topology to accomplish cluster reorganization (Message Complexity).

The notion of upstream member, downstream member, and peer member are defined as follows. Given node j and node k are members of the same cluster, node j is the upstream member of node k if $h_j < h_k$; node j is downstream member of node k if $h_j > h_k$; node j is peer member of node k if $h_j = h_k$. These three cases are illustrated in Figure 4.15 as Case 1, Case 2 and Case 3 respectively.

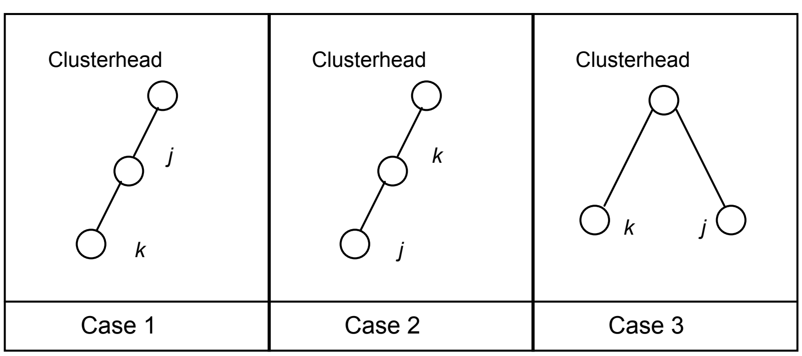


Figure 4.15 Notion of upstream, downstream and peer member in MobDHop.

4.4.3 Hello Protocol Overhead

The hello messages are broadcasted for every predefined Hello interval during the communication session for nodes to learn its neighbourhood and corresponding variability information in order to compute local variability value which will be used in clusterhead election. Therefore, each node emits a certain amount of Hello messages per time step in order to maintain up-to-date neighbourhood knowledge. This incurs an overhead of $r_{hello}N$ packets per time step. Since r_{hello} is a constant predefined by the protocol and the communication session consists of D time steps, ψ_{H} is O(DN).

4.4.4 Cluster Formation Overhead and Time Complexity

In the first phase of MobDHop (i.e. one-hop cluster formation), each node will first measure relative mobility with respect to all neighbours for a predefined sampling period, T_{sample}. Hence, the number of time steps each node takes before it can decide to be a clusterhead or to join a neighbouring cluster is at least T_{sample}. After the clustering decision is made, each node will broadcast a new Hello message with its latest cluster decision. If a node opts to be a clusterhead, it will broadcast a Hello message to its neighbours that contain its cluster ID and group variability. This is a trivial case and takes only 1 time step. On the contrary, if a node opts to join a neighbouring cluster, it will broadcast its decision to its clusterhead which is at most one hop away from the node. Therefore, this message takes at most 1 time step to reach the clusterhead. In short, time complexity of cluster formation in MobDHop is $T \leq T_{sample} + 1$. Message Complexity, on the other hand is $M = 1$. Since the cluster formation process will only occur once during the cluster setup phase, the cluster formation overhead, $\psi_{\mathrm{CF}} = \mathrm{O}(T_{sample}N)$.

4.4.5 Cluster Maintenance Overhead and Time Complexity

Cluster maintenance in MobDHop is done by continuous inspection on local information via periodical messaging. The approach used to analyze overhead required by cluster maintenance process is greatly inspired by the analysis of DMAC done by Bettstetter and Krausser [132]. In MobDHop, if a topology change is detected, the node will take respective action to maintain the cluster structure. There are three types of events that may cause a topology change in MANETs:

i. A node joins the network.

ii. Two nodes move away from each other transmission range (link failure).

iii. Two nodes move into each other transmission range (link establishment).

4.4.5.1 Joining of New Node

After a new node joins the network, it has to make clustering decision, i.e. to decide which cluster to join and what role to play. This process is determined by two factors:

i. The state of nodes in its neighbourhood.

ii. Relative mobility with respect to every neighbour.

In MobDHop, a new node, say node *a*, will first try to merge into neighbouring clusters by measuring its relative mobility with respect to each neighbour for T_{sample} time steps and compute their variation of ED over time. It will choose the neighbour which is relatively most stable, i.e. it yields lowest variation of ED with respect to that neighbour. Denote the neighbour as node *b*. If node *b* is connected to its clusterhead by an unsaturated link (i.e. link which may consist of multiple hops but the hop count is less than H_{max} hops), node *a* joins the cluster successfully. If this condition fails, node a will decide its role (clusterhead or ordinary node) by taking all non-clustered neighbours into consideration during a clusterhead election as in cluster setup phase. Therefore, the message and time complexity depend on the configuration of neighbourhood at the time when the topology change occurs.

We denote the number of neighbours of a node, i.e. its degree, as *deg*. Four kinds of neighbours are identified, i.e.:

i. Neighbours that are clusterhead (d_{ch}).

ii. Neighbours that have joined a cluster and are connected to their clusterhead by an unsaturated link ($d_{us\text{-}mem}$).

iii. Neighbours that have joined a cluster and are connected to their clusterhead by a saturated link ($d_{s\text{-}mem}$).

iv. Neighbours that are still not clustered to any cluster (d_{nc}).

Therefore, the total number of neighbours of a node, $deg = d_{ch} + d_{us\text{-}mem} + d_{s\text{-}mem} + d_{nc}$. If a new node, *i* has no neighbours (d = 0), a trivial case occurs. It selects itself as clusterhead and broadcasts its decision in the next Hello message (M = 1). This process is done in one time step (T = 1).

If node *i* has at least one neighbour that is clusterhead, or cluster member that is having an unsaturated link ($d_{ch} + d_{us\text{-}mem} > 0$), node *i* will start to collect information for local variability computation and decide its cluster membership after T_{sample} time steps. After making the decision, it will propagate this decision to its new clusterhead. The time needed for this decision to arrive is at most H_{max} time steps since the clusterhead is at most H_{max} away. Therefore, $T \le T_{sample} + H_{max}$ and $M \le H_{max}$.

Table 4.5 Time and message complexity due to different neighbourhood configuration.

Neighbourhood Configuration				Complexities		New Status
d_{ch}	$d_{us\text{-}mem}$	$d_{s\text{-}mem}$	d_{nc}	T	M	
0	0	0	0	1	1	CH*
> 0	0	any	any	$\le T_{sample} + 1$	1	ON**
0	> 0	any	any	$\le T_{sample} + H_{max}$	$\le H_{max}$	ON
0	0	0	> 0	$\le T_{sample} + H_{max}$	1	CH / ON

* CH denotes ClusterHead; ** ON denoted Ordinary Node

In the third case where all neighbours nodes are not yet clustered ($d_{nc} > 0$), node *i* will perform a similar process as in cluster formation phase. Therefore, the time and message complexity is the same as those in cluster formation. Since MobDHop adopts Least Clusterhead Change (LCC) mechanism during cluster maintenance, chain reaction caused by any cluster reorganization can be avoided. The time and message complexities for different kinds of neighbourhood configurations are summarized in Table 4.5.

4.4.5.2 Link Failure

A link failure between nodes from two different clusters or between any two ordinary peer member nodes will not cause any cluster reorganization in MobDHop. Only link failure between an ordinary node and its clusterhead or its upstream ordinary node will trigger the cluster reorganization process. In both cases, only downstream member node will react to this topology change since clusterhead or upstream ordinary node will simply eliminate downstream member nodes from their member lists. The reacting downstream member node is denoted as node *a*. First, we consider a base case when node *a* is a border node, i.e. it has no downstream members. Three similar cases may happen as in previous section where a new node is added into network. Therefore, time and message complexity are the same as in new node scenario (cf: Table 4.5).

In another case when the reacting node has downstream members, each downstream member has to react when they receive messages from their upstream member about status or cluster membership changes. Therefore, this is a chain reaction, which will be reaching an end when the effect reaches the border node of the cluster where the above mentioned base case is executed. Then, cluster reorganization is complete and a valid cluster structure is re-established. In other words, the chain reaction can at most propagate to (H_{max} - h_a + 1) hops. The total number of nodes that will be affected by this topology change is the number of downstream members, which is less than the number of members in the cluster. Therefore, time complexity and message complexity is upper bounded as shown below:

- $T \leq (T_{sample} + H_{max}) + (H_{max} - h_a + 1)$
- $M \leq mH_{max}$

4.4.5.3 Link Establishment

A link establishment between two ordinary nodes will not cause any cluster reorganization since both node is still connected to their clusterheads. In case a new link is established between an ordinary node and a clusterhead, no cluster reorganization shall take place since the cluster structure is still valid. When a new link is established between two clusterheads, clusterhead contention occurs and MobDHop will resolve the clusterhead contention by making the clusterhead, which is less stable (higher group variability value) to give up its role and join the winner cluster as an ordinary node. If the loser has no members at all, the cluster reorganization is complete. Therefore, the loser node broadcasts its decision in the next Hello message ($M = 1$) and the process is completed in one time step ($T = 1$). Otherwise, all members are subject to cluster reorganization. A similar process as in link failure case will be carried out. The base case occurs when the reacting node *a*, is a border node. Three possible cases could happen as in previous sections, i.e. link failure and new node scenario. Therefore, time and message complexity for base case are the same as shown in Table 4.5.

If the reacting node has downstream members, each downstream member has to react when they receive Hello messages, indicating clusterhead or status changes. Again, this is a chain reaction that will come to an end when the effect reaches the border node of the loser cluster. Since each member is at most H_{max} hops away from its clusterhead, chain reaction will at most extend to H_{max} hops and may involve all cluster members. In short, the upper bounds of message and time complexity after a link establishment event are listed as below:

- $T \leq (T_{sample} + H_{max}) + (H_{max})$
- $M \leq mH_{max}$

4.4.5.4 Total Cluster Maintenance Overhead

As analysed in Section 4.4.5.1, 4.4.5.2, and 4.4.5.3, the upper bound of message complexity is $M = mH_{max}$ per topology change. To quantify the topology change, the results from Sucec and Marsic [137] are adopted. Sucec and Marsic [137] presented a detailed analysis on the average number of link state change events per time step based on Random Waypoint mobility model. According to this paper, average number of link state change events, i.e. topology changes, per time step is given as:

$$r_{link} = \Theta\left(\frac{\mu}{R} \bullet |E|\right) = \Theta\left(\frac{\mu}{R} \bullet N \bullet \frac{d}{2}\right) = \Theta(N)$$

Therefore, the average number of topology changes in the network grows asymptotically with the number of nodes in the network. $H_{max} < d$ and d is a constant predefined in the algorithm to limit the diameter of cluster formed. Therefore $M = O(m)$ per topology change. The cluster maintenance overhead, $\psi_{CM} = O(mND)$.

4.4.6 Total MobDHop Clustering Overhead

Our analysis is summarized in Table 4.6. The total clustering overhead, ψ_C, is the sum of the following three factors:

i. Hello Protocol Overhead (ψ_H)

ii. Cluster Formation Overhead (ψ_{CF})

iii. Cluster Maintenance Overhead (ψ_{CM})

Therefore, the total clustering overhead incurred by MobDHop clustering algorithm is $O(DN)$ + $O(T_{sample}N)$ + $O(mND)$. Dividing this results by D time steps, the total clustering overhead is $O(N)$ + $O(N)$ + $O(mN)$ in the network. Dividing this result by node count N yields total MobDHop clustering overhead, $\psi_C = O(m)$ per node per time step. Since m is the average number of members in a cluster. It is always smaller than network size. It is also feasible to add a parameter in order to limit the size of each cluster so that the cluster size formed by MobDHop is constrained to a constant value. Thus, the total clustering overhead of MobDHop can be constrained to $\psi_C = O(1)$ per node per time step as per **Claim 1**.

Table 4.6 Summary of overhead and time complexity analysis on MobDHop

Overhead Type	Time Complexity	Message Complexity	Total Overhead per time step
Hello Protocol	1	$< r_{hello}N$	$O(N)$
Cluster Formation (per topology change)	$< T_{sample} + 1$	$< N$	$O(N)$
Cluster Maintenance (per topology change)	$< (T_{sample} + H_{max}) + (H_{max})$	$< mH_{max}$	$O(N)$

4.4.7 Analysis Verification via Simulations

Simulations were performed using QualNet Simulator 3.8 to investigate the message complexity of MobDHop in the presence of mobility. Maximum cluster size constraint was not imposed in these simulations. The overhead incurred by the Hello protocol was not taken into account in this simulation study because the amount of signalling overhead incurred by the Hello protocol is O(1) per node per time step. Furthermore, most of the existing clustering algorithms such as Lowest-ID, MCC, and MOBIC assume a Hello protocol in place. Hello protocol is also widely used in routing protocols as a neighbourhood discovery mechanism [138]. The additional signalling overhead incurred by forming and maintaining multihop clusters using MobDHop is the main concern of this section. There are two types of control packets in this MobDHop implementation, i.e. Join-Packet and Leave-Packet. Join-Packet and Leave-Packet are sent to the clusterhead whenever a node joins or leaves a clusterhead which is more than one hop away. The broadcast nature of the wireless medium allows one-hop neighbours to join and leave the cluster implicitly by tagging some additional fields in Hello messages. Since the first phase of MobDHop forms one-hop clusters, no additional control packets are needed. Therefore, the only MobDHop overhead is the cluster maintenance overhead as discussed in Section 4.4.5.4.

In these simulations, the RW mobility model was assumed. Each simulation was executed for 900 seconds. *d* was set to 2 for all simulations. This value was chosen by considering the simulated network size. If a larger value is chosen, MobDHop will form less but larger multihop clusters since a random mobility model was assumed. Transmission range was homogeneous for every node, i.e. R = 376 meters, which is the default value for IEEE 802.11 DCF with channel capacity of 2Mbps in QualNet simulator. In the first set of simulations, average node speed was varied while network density was fixed. Each scenario consisted of 50 nodes that were moving continuously in a 3000m x 3000m area. In the second set of simulations, network size was varied from 50 to 600 nodes (cf: Table 4.7) while the average node speed and the network density were held as constant. Average node speed was fixed at 12 m/s in all scenarios.

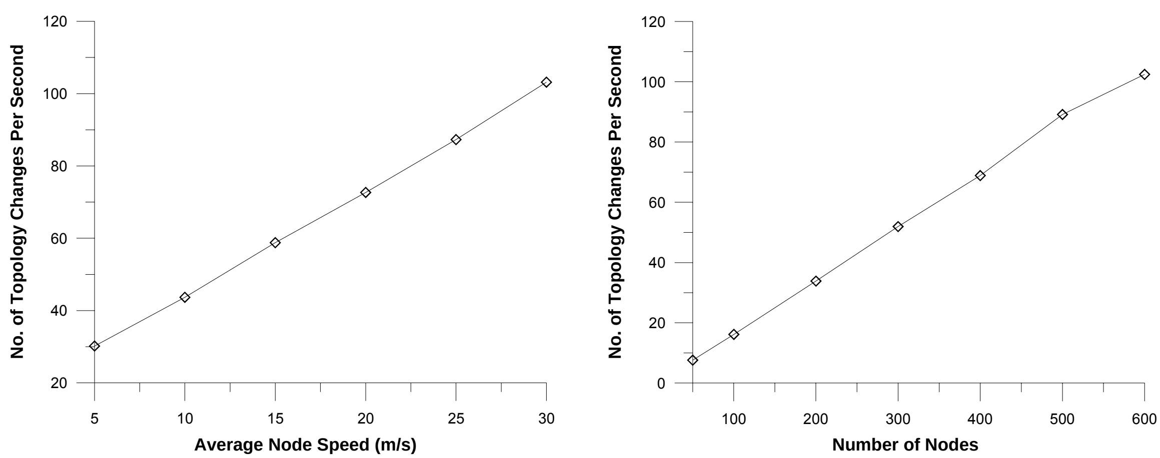
Table 4.7 Varying network size (constant network density)

Number of Nodes	Area (m^2)	Average Number of Neighbors
50	2000 x 2000	6.97
100	2850 x 2850	7.00
200	4000 x 4000	7.08
300	5000 x 5000	7.19
400	5800 x 5800	7.11
500	6500 x 6500	7.18
600	7250 x 7250	6.93

Figure 4.16 shows the topology change rate increased with the average node speed and the number of nodes in the network. These results confirmed the analysis in [137] that the topology change rate under Random Waypoint mobility model is influenced by average node speed and the number of nodes in the network. To evaluate the percentage of topology changes that actually causes clustering overhead, the ratio of the number of topology changes that cause cluster structural changes to the total number of topology changes in the network was measured. This ratio is named Effective Topology Change (ETC) ratio. As shown in Figure 4.17(a), ETC ratio in the first set of simulations (varying average node speed) varied negligibly in the range of 0.3 and 0.4. In the second set of simulations (varying network size), ETC ratio also varied negligibly in the

range of 0.25 and 0.3 as shown in Figure 4.17(b). This implies that, in terms of clustering overhead, MobDHop is less sensitive with respect to both mobility rate and network size. Since MobDHop is a mobility-adaptive clustering algorithm that forms clusters which are as stable as possible, the clustering overhead caused by cluster changes due to mobility can be kept to minimum in MobDHop.

Figure 4.18(a) shows the number of control packets per node increased with the average node speed. Since a constant number of control packets will be incurred with a topology change, an increase in the number of control packets with average node speed is anticipated. Figure 4.19(a) shows the number of control packet per node remained constant in the second set of simulations. This is because the network density and average node speed were fixed in these simulations. Therefore, the number of topology changes experienced by each node was similar. Meanwhile, Figure 4.18(b) and Figure 4.19(b) show the number of control packets per effective topology change did not vary much in both sets of simulations. Effective topology change is the topology change that causes at least one cluster structural change. Our simulation results show that MobDHop incurs a consistent amount of control overhead per cluster structural change. This is consistent with **Claim 1** in our previous theoretical analysis.



(a) Topology Change Rate vs Average Node Speed **(b) Topology Change Rate vs Node Number**

Figure 4.16 Impact of average node speed and network size on the topology change rate.

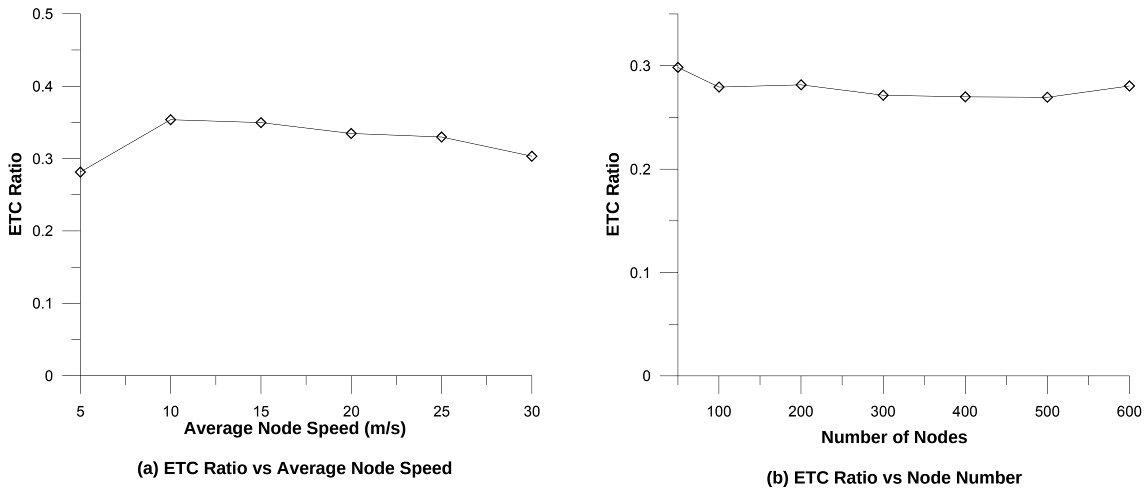


Figure 4.17 **Impact of average node speed and network size on the effective topology change ratio.**

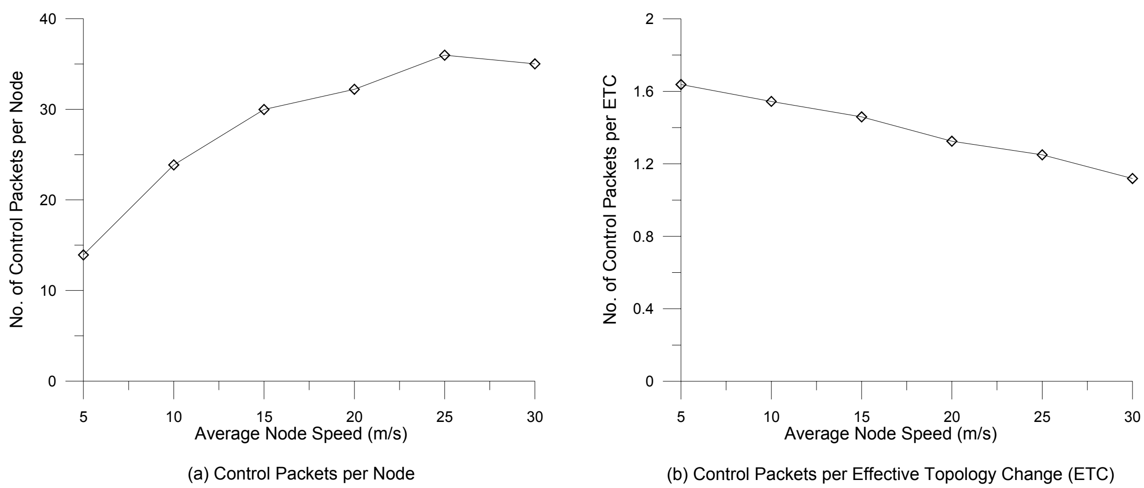


Figure 4.18 **The impact of average node speed on the MobDhop clustering overhead.**

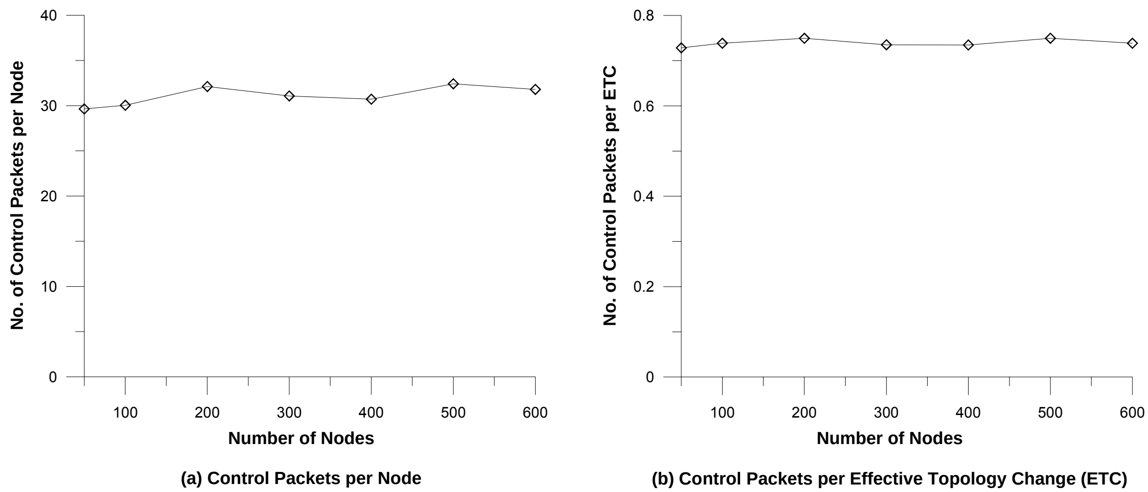


Figure 4.19 The impact of network size on the MobDHop clustering overhead.

4.4.8 Comparison of Clustering Overhead by Five Clustering Algorithms

The theoretical analysis of the time and message complexity of Lowest-ID and MCC clustering algorithm is similar to the analysis of DMAC [44]. However, we assume the LCC improvement is applied on both Lowest-ID and MCC clustering algorithms. Therefore, a tight bound on the time and message complexity can be derived. The results are presented in Table 4.8. Table 4.8 summarizes the comparison of the time and message complexity among MobDHop, Max-min *d*-Clustering, Lowest-ID, MCC and DMAC. For DMAC, the worst case lower bound for the time complexity is given in reference [44] during a new node event where all neighbours are either clusterheads with lower weight or ordinary nodes with lower weight.

The overhead of the hello messaging, OH_H is $O(N)$ per time step. The total overhead incurred during cluster formation, OH_{CF} is $O(N)$ per time step too. Each topological change will incur at most O(1) overhead. Therefore, the average number of link state change events based on the random mobility model is $\Theta(N)$. Hence, the total cluster maintenance overhead per time step is given by $OH_{CM} = O(N)$. The total clustering overhead incurred by Lowest-ID or MCC is $O(N) + O(N) + O(N) = O(N)$ per time step. Dividing this by the number of nodes yields a total clustering overhead, $OH_C = O(1)$ per node per time step.

The Distributed Mobility-Adaptive Clustering (DMAC) is similar to the Lowest-ID and MCC clustering algorithms except for the clusterhead election criteria. The role of a node is determined by a weight which is associated with every node based on some predefined criteria, e.g remaining power, speed and node ID. Bettstetter and Konig [44] investigated the reaction of the DMAC algorithm towards topology changes in a network and analysed the message and time complexity of this algorithm. They observed the inevitable reclustering chain reactions which are resulted from topology changes e.g. node addition, link failure and link establishment. Reclustering chain reaction may happen when a topology change involves a clusterhead with certain neighbourhood configurations that may lead to another clusterhead in its neighbourhood to give up its role. The effect of chain reactions is unpredictable and therefore only lower bounds for time and message complexity were provided. The worst case happens when a chain reaction occurs where a valid cluster structure takes at least 2 time steps to be formed with a message complexity of $1 + deg$.

Max-min d-Clustering is a heuristic to form d-hop clusters in MANETs. Each node is at most d hops away from its clusterhead. The heuristic can be executed at regular intervals or whenever there is a topology change to maintain valid cluster structure. The heuristic is claimed to elect fewer clusterheads and form larger clusters with longer clusterhead duration on the average than the Lowest-ID algorithm. However, Max-min d-Clustering does not take mobility pattern into account during cluster formation. The time complexity of Max-min d-Clustering [50] is $O(2d + d)$. Whenever the heuristics is executed, each node has to send at least d messages before a cluster can be formed. Since the average number of link state change events, i.e. topology changes, per time step is $\Theta(N)$, the total overhead per time step incurred by Max-min d-Clustering is $O(dN)$. Since d is a predefined constant in Max-Min D algorithm, the total overhead per time step is also $O(N)$.

Table 4.8 Comparison among five different clustering algorithms

Algorithm	*Time Complexity per topology change*	*Message Complexity per topology change*	**Total Overhead per time step**
MobDHop	$\leq(T_{sample} + H_{max}) + (H_{max})$	$\leq mH_{max}$	O(N)
Max-Min	O(d)	O(d)	O(N)
Lowest-ID	O(1)	O(1)	O(N)
MCC	O(1)	O(1)	O(N)
DMAC	$\geq 1 + deg$	≥ 2	O(N)

Our analysis shows that the total clustering overhead of one-hop clustering or multihop clustering are similar in the asymptotic upper bound with respect to the number of nodes in network. MobDHop, Lowest-ID and MCC have a better time complexity than DMAC because the re-clustering chain reaction is avoided by LCC improvement. LCC improvement provides a better performance in terms of message complexity. LCC improvement is integrated into MobDHop, Lowest-ID and MCC clustering to avoid re-clustering chain reactions. Still, chain reaction may occur in MobDHop clusters but it is restricted to H_{max} hops. LCC was not integrated into DMAC in this thesis since LCC will force the second property of a valid cluster structure given by Basagni [75] to be violated, i.e. every ordinary node affiliates with the neighbouring clusterhead with the bigger weight.

4.5 Unicast Performance using MobDHop

In this section, we investigate the use of MobDHop clustering algorithm to provide an underlying cluster structure for unicast routing protocol in MANETs. A new variant of AODV, namely MobDHop-AODV, is introduced to work on top of the stable, two-tier cluster structure formed by MobDHop clustering algorithm. The performance of MobDHop-AODV was evaluated using simulations and compared with the original AODV [139].

4.5.1 Protocol Operation

To investigate the effectiveness of the cluster structure provided by MobDHop algorithm, a cluster-based unicast routing protocol based on the AODV [139], namely MobDHop-AODV, was developed and tested using the QualNet commercial simulator. The goal of this protocol is to exploit the aggregated topology information stored at every clusterhead to avoid the need to flood the network with route request (RREQ) packets in the search for intended destinations.

The AODV routing protocol is a reactive unicast routing protocol that constructs and maintains unicast routes in MANETs. It avoids routing loops by introducing the use of sequence numbers. There are three types of control messages used by AODV: Route Request (RREQ) messages are initiated from the source node when it needs to send data to a destination node which it does not have a valid or existing path. Each node that receives the broadcasted RREQ message will update its routing table with the knowledge of route to source node. Route Reply (RREP) messages will be initiated by either the target node or intermediate nodes if the latter has a valid route to the destination that is "fresh enough", based on the sequence numbers. Route Error (RERR) messages are used to notify the other nodes which use routes that have broken links. Link connectivity information is maintained by periodical broadcast of Hello messages.

In MobDHop-AODV, two extra protocol messages are introduced. Cluster Request (CREQ) messages are initiated from the source node when it needs to send data and the route to destination is still unknown. The source node first unicasts a CREQ message to its clusterhead. Clusterhead, upon receiving CREQ message, will check its membership table for the destination node. Cluster Reply (CREP) will then be sent by the clusterhead back to the source. If the destination node is found in the cluster, a Boolean flag, namely *InCluster* flag in CREP message is set to true. At the same time, clusterhead will initiate a RREQ packet to destination node in order to set up path. If the destination node is not found in the cluster, the *InCluster* flag is set to false. Upon receiving CREP message, the source will check the value of the *InCluster* flag. If the flag indicates a true value, the source node will transmit data packets by using the path set up during the propagation of

CREQ message. Otherwise, the source node will initiate a network-wide flooding of RREQ message to search for the route to destination node. This additional routine will reduce the number of network-wide RREQ messages initiated by the source node if both the source and destinations nodes belong to the same cluster. The possibility of having broadcast storms [88] can be reduced and the limited resources such as channel resources and device resources in MANETs can be preserved.

4.5.2 Simulation Environment

Simulations were conducted by using Qualnet 3.8. The communication range is 376 metres which is the default value for IEEE 802.11 DCF with channel capacity of 2Mbps in Qualnet. For each network configuration, ten different scenarios were generated by randomizing the seed value and each data point was therefore the average of 10 simulation runs. In these simulations, 200 nodes were simulated over an area of 2000 metres by 2000 metres. Nodes moved according to the RPGM model. The average node speed was varied from 0m/s to 20m/s. Each mobility group consisted of 20 nodes and the maximum group deviation distance was set to 500 metres. The duration of each simulation was 600 seconds. All MobDHop parameters were similar to those used in Table 4.4. Each source starts to generate Constant Bit Rate (CBR) traffic at the rate of two 512-bytes data packets per second for 300 seconds. The starting instances were randomly chosen between 0-300 seconds. We simulated two scenarios which consisted of 20 and 30 connections respectively.

The performance of MobDHop-AODV and the original AODV protocol were evaluated based on the following metrics:

i. *Packet Delivery Ratio*: The number of data packets successfully delivered to destinations over the number of data packets should be delivered to destinations.

ii. *Number of RREQ Transmitted*: The total number of RREQ messages transmitted by source and intermediate nodes.

iii. *Number of Routing Control Packets Transmitted*: The total number of control packets transmitted by source, destination and intermediate nodes for unicast routing purpose. For AODV, these control packets include RREQ, RREP and RERR. For MobDHop-AODV, these control packets include RREQ, RREP, RERR, CREQ and CREP.

iv. *Average End-to-End Delay*: The average duration from the time at which a data packet is generated and the time at which it is received by the destination.

4.5.3 Simulation Results and Discussions

Figure 4.20 and Figure 4.21 show the performance of MobDHop-AODV and AODV with respect to the increase in average node speed for 20 and 30 connections recpectively. Packet delivery ratio of MobDHop-AODV, as shown in Figure 4.20(a) and Figure 4.21(a), decreased with the increase in average node speed as the topology is more dynamic in the network of higher mobility rate. The packet delivery ratio of MobDHop-AODV was comparable to the packet delivery ratio of the original AODV. As depicted in Figure 4.20(a) and Figure 4.21(a), the packet delivery ratio of AODV in static network (network with 0m/s average node speed) was slightly lower. Since there is no mobility in static network, the considerable packet loss in AODV could only be due to the serious contention and collisions at MAC layer between data and control packets. It was observed in Figure 4.20(c) that the routing control packets incurred by AODV in static networks was about three times higher than those incurred by MobDHop-AODV.

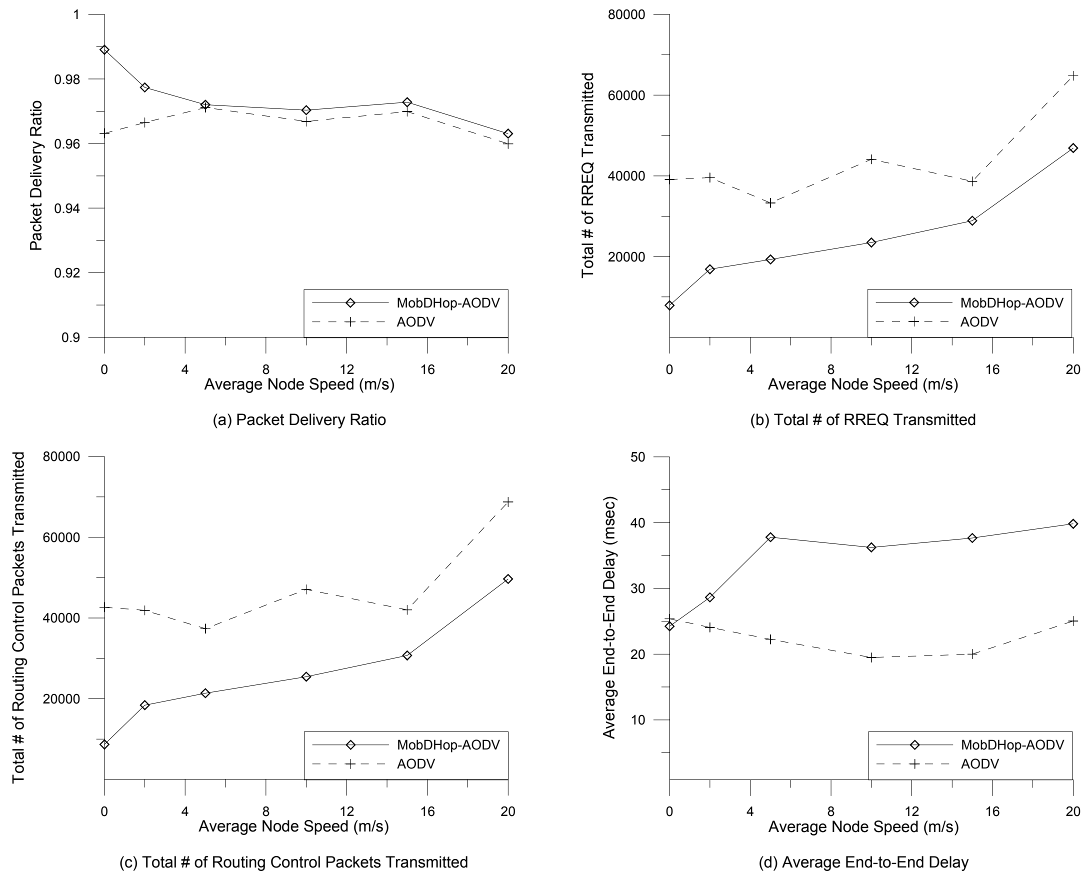


Figure 4.20 **The performance of MobDHop-AODV and original AODV vs. the increase in node speed with 20 connections.**

Figure 4.20(b) and Figure 4.21(b) show that MobDHop-AODV successfully reduced the total number of network-wide RREQ messages sent by 20-75% over scenarios of different speeds and traffic load. This is attributed to the fact that MobDHop-AODV introduces an extra routine that requires source node to make a query to its clusterhead for the location of destination before initiating a network-wide flooding of RREQ messages. If the destination is located inside the same cluster as source node, unnecessary network-wide floodings of RREQ message can be avoided. MobDHop-AODV also transmitted much fewer routing control packets over the network of different average node speed as shown in Figure 4.20(c) and Figure 4.21(c). The significant reduction in routing overhead can increase the ability of MANET to support more unicast traffic or other types of traffic in the network. Due to the additional intra-cluster request cycle in MobDHop-

AODV, extra latency will be incurred during the route discovery phase. It is shown in Figure 4.19(d) and Figure 4.21(d) that MobDHop-AODV incurred higher average end-to-end delay because of the additional clusterhead query routine. The impact of extra latency introduced by MobDHop is less significant in static networks (networks with 0m/s average node speed) as the routes in static networks seldom break. Therefore, there is no need to initiate frequent route query, either intra-cluster CREQ or network-wide RREQ to search for destinations.

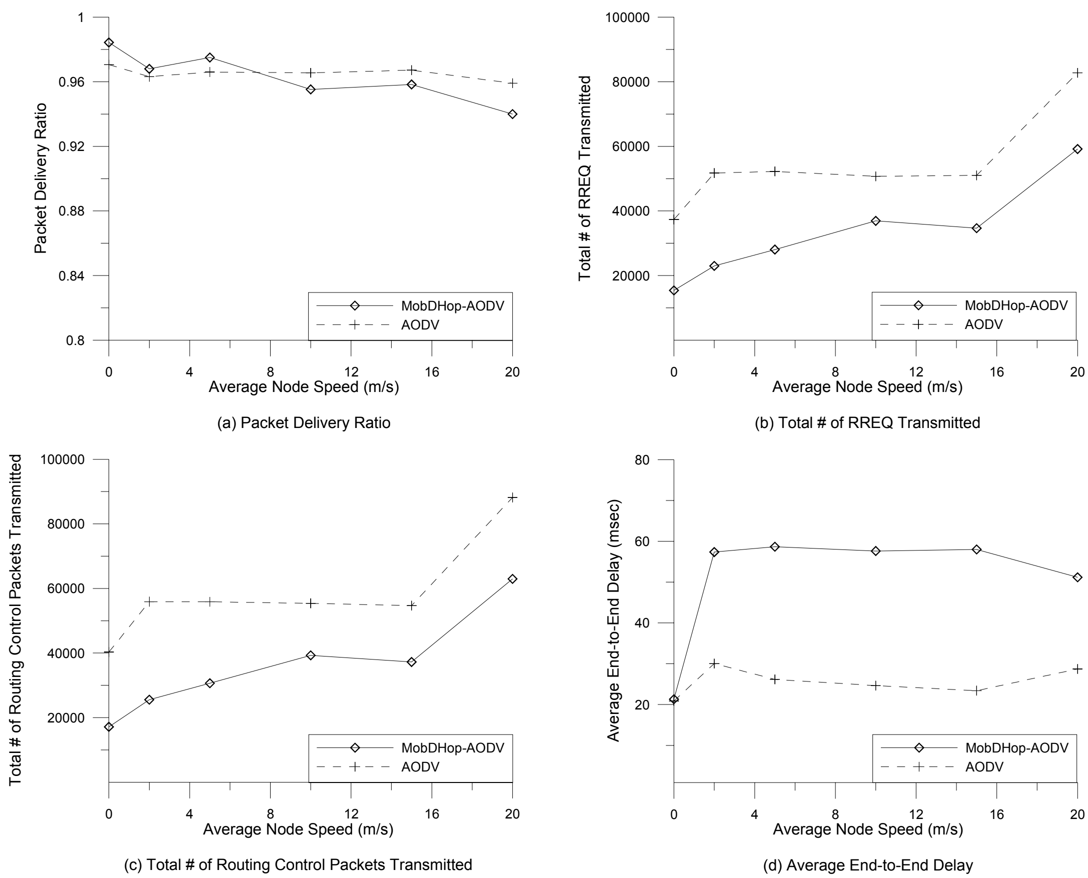


Figure 4.21 The performance of MobDHop-AODV and original AODV vs. the increase in node speed with 30 connections.

4.6 Summary

In this chapter, both empirical and theoretical methodologies are adopted in the performance evaluation of MobDHop. Empirically, network simulations were conducted in the widely-used

NS-2 and QualNet simulators. Simulation results show that MobDHop outperforms L-ID and MCC algorithms in both RW and RPGM models in terms of clustering efficiency and cluster stability. Results also show that MobDHop is beneficial for different kinds of group communication in large, sparse MANETs. Furthermore, the performance of MobDHop in small, dense MANET is comparable if not better than existing clustering algorithms.

Unlike other multihop clustering algorithms, *d* can be set to a very large value in MobDHop. Even if *d* is larger than the network diameter, MobDHop will not form unreasonably large clusters as in other multihop clustering algorithms. *d* is an important parameter in other multihop clustering algorithms that must be defined before the execution of the algorithm in order to limit the multihop clusters from growing too large. This is not the case in MobDHop since MobDHop uses cluster stability information to determine the diameter of stable multihop clusters while *d* is primarily a limiting factor that can be set to control the network from growing too large based on the network management requirements. When the stability criterion is not met during the merging phase, cluster will not grow and remain in its most stable state. Hence, MobDHop can adaptively form variable-hop clusters which are more stable based on the use of local variability metric to identify the mobility patterns in the MANETs.

The analysis of message and time complexity of MobDHop provides insights into how MobDHop reacts to network topology changes. We claim that the number of packet transmissions per node per time step required for MobDHop to operate correctly in MANETs is O(1). The upper bound of time complexities for both cluster formation and cluster maintenance in MobDHop are provided. Our claim is verified via network simulations.

It was also shown in this chapter that MobDHop can support unicast routing functionality by integrating MobDHop into a well-known unicast routing protocol, AODV. A new intra-cluster query routine is introduced into AODV to exploit the knowledge of clusterheads elected by MobDHop algorithm. This new variant of AODV is named MobDHop-AODV. The goal of MobDHop-AODV is to reduce network-wide flooding of RREQ messages. Simulation results showed that the number of network-wide RREQ messages was successfully reduced for about 20-80% in networks of different speeds.

In short, stable multihop clustering is demonstrated in this chapter to be feasible and practical in ad hoc networks of high mobility rate without incurring prohibitive signalling overhead. This stable multihop clustering can be used to form stable two-tier cluster structure to support various network control function such as unicast routing, multicast routing, location management, data replication and so on.

CHAPTER 5

CLUSTER-BASED, GROUP-ADAPTIVE MULTICAST ROUTING PROTOCOL

5.1 Introduction

As discussed in Chapter 2, existing multicast routing protocols can be generally categorized into tree-based and mesh-based scheme based on their multicast forwarding infrastructure. Tree-based schemes, similar to those used in IP multicasting, such as MAODV, AMRoute, AMRIS, and ADMR, were proposed to support multicast routing in MANETs. However, frequent link breaks cause considerable changes in tree-based structures and, packet loss is inevitable during the recovery process since each destination is connected to the tree by single path. In view of this, mesh-based schemes, such as ODMRP and CAMP, were proposed to provide redundant paths for forwarding multicast packets, but packet loss is reduced at the cost of increased data overhead. It is suggested in [27] that a simple broadcast scheme is the most reliable and feasible solution in highly mobile MANETs. However, it is obvious that the main drawback of mesh-based and broadcast scheme is the excessive consumption of the network resources due to a large amount of redundant data packets. Futhermore all these schemes have been designed with small networks in mind. Hence most of the simulations used to validate these schemes featured small-scale MANETs. The performances of these schemes in large MANETs which may consist of a large number of nodes and stretch across a large physical area remain unclear. Moreover, a flat routing philosophy is adopted in most of the multicast routing protocols proposed. Some protocols propose variations of the basic route discovery and maintenance techniques in order to improve their scalability such as core-assisted member joining and expanding ring search (ERS) in CAMP as well as scoped flooding for localized route maintenance in NSMP. However, they may not scale well to large networks. Recently, a shift towards protocol state reduction to support protocol scalability in the design of multicast routing protocol for MANETs is observed in the proposal of hierarchical [115] and stateless multicasting [95].

In this thesis, a two-tier multicast routing protocol for MANET is proposed with the goal to provide better protocol scalability in terms of a set of network parameters like network density, network size, traffic load, mobility, and multicast-related parameters, and at the same time not compromising the protocol robustness and multicast efficiency. Group-AdaPtivE (hereafter known as GRAPE) multicast routing protocol, which works on top of a pre-existing two-tier cluster structure, is proposed.

This work is motivated by two observations. First, forming a stable two-tier cluster structure is possible in MANETs with high mobility. Node mobility pattern is mainly determined by the nature of applications, and since mobile devices are usually carried by or associated with humans, the movement of such devices is necessarily based on human decisions and socialization behaviour. Mobile users are likely to exhibit correlated mobility patterns in their movements, which is also known as group mobility. The validity of this assumption is further strengthened by the collaborative nature of typical MANET applications such as disaster relief operations, battlefield operation, and conference scenarios. The nodes do not behave randomly but they are usually involved in team activities to achieve common goals. This group mobility pattern enables the formation and identification of stable cluster structures in these MANETs via appropriate clustering scheme. The cluster structure could serve as the routing architecture for MANETs in order to implement a scalable hierarchical multicast routing protocol. Existing tree-based and mesh-based schemes do not take the group mobility into consideration during the formation of their multicast forwarding infrastructure. Second, different group communication patterns may co-exist in a team. For example, each team leader may join a multicast group that is formed to propagate important instructions from the commander in battlefields. At the same time, each team leader may initiate another multicast group within its team to propagate his strategies to his soldiers. It is observed that previously proposed multicast routing schemes do not take the traffic pattern in the application layer into consideration. This piece of information could be useful in order to facilitate efficient multicast routing and reduce both the control and data overhead incurred in multicast routing.

In GRAPE, a two-tier non-overlapping cluster structure is assumed and the diameter of each cluster should be flexible and dependent on the mobility pattern in the networks. Each cluster is led by a clusterhead which is usually located in the middle of the cluster to exploit the "wireless broadcast advantage" [140]. Clusterheads are responsible for: 1) representing their cluster members in joining the multicast group session based on the interest of their cluster members and 2) switching adaptively between two multicast strategies, i.e. (a) cluster broadcasting and (b) stateless tree-based multicasting, to deliver data packets to relevant multicast group members in their cluster based on traffic characteristic within the cluster.

The main advantage of this approach is the exploitation of two-tier cluster structure in order to achieve reduced protocol state maintenance overhead and better protocol scalability in terms of a set of network parameters. By allowing the clusterhead to represent its cluster in group communication, GRAPE significantly reduces the number of nodes that participate in the construction and maintenance of the upper-tier multicast forwarding infrastructure and thus drastically lowers protocol overhead. Besides, the protocol adaptability to multicast property also reduces unnecessary data overheads significantly. The adaptation to multicast property within a cluster enables the use of broadcasting when a large number of cluster members are interested in the multicast communication. Therefore, the "wireless broadcast advantage" is maximized in the cluster and this local broadcast also provides more robustness against node mobility. Conversely, a simple tree-based scheme is used when only a relatively smaller number of cluster members are interested in the multicast communication. This can save bandwidth by avoiding unnecessary broadcasts of data packets especially in relatively larger multihop clusters. A detailed description on the operation of GRAPE will be presented in the following sections.

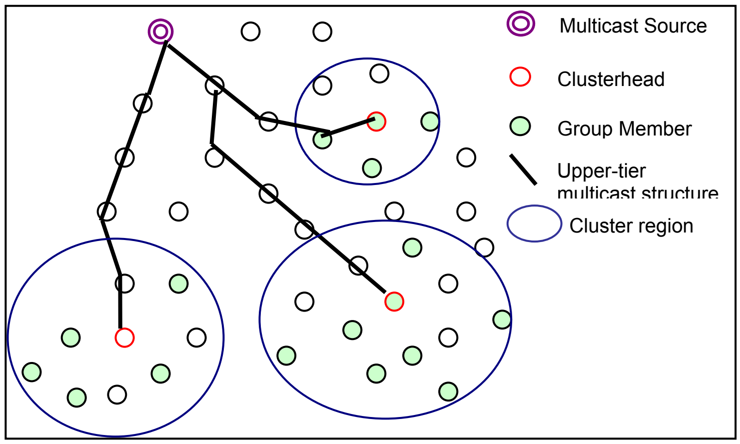


Figure 5.1 Grape-like two-tier multicast hierarchy.

5.2 GRAPE Multicast Routing Protocol

GRAPE forms a grape-like group communication structures for data packet delivery as shown in Figure 5.1. More specifically, a two-tier hierarchical structure is formed (see Figure 5.1) where the upper tier is formed by multicast sources and clusterheads that join the multicast group communication and the lower tier consists of cluster members that are interested in the multicast communication and their respective clusterhead. In the following sections, details of the construction of the two-tier cluster structure, a new hierarchical multicast group management scheme and multicast packet forwarding mechanism in GRAPE will be presented.

5.2.1 Protocol Messages and Data Structures

GRAPE requires four types of control messages for multicast routing, i.e. Multicast-route-REQuest (hereafter known as **MREQ)**, Multicast-route-REPly (hereafter known as **MREP)**, and

Multicast-Member-Join-or-Leave (hereafter known as **MemberJL)**. MREQ is sent by every source node periodically to refresh both group membership and multicast delivery infrastructure. MREP is sent by every clusterhead which intends to join multicast session upon receiving MREQ from the relevant multicast source. MemberJL is sent by cluster members to their respective clusterhead to express their interest to join or leave a particular multicast group. If a member intends to join a multicast group, the join flag is set to true. Otherwise it is set to false. For easy reference, we refer the MemberJL packet with join flag set to true as **MemberJoin** packet. Otherwise, it is referred as **MemberLeave** packet.

Every multicast source will maintain a clusterhead membership table, i.e. **MG-Membership-Table** in order to keep track of the list of clusterhead addresses that have joined the multicast session. Each clusterhead, on the other hand, maintains a table, i.e. **Cluster-Membership-Table** to keep track of the list of cluster members joining different multicast groups. Each node in the network maintains two data structures: a **MG-Flag-Cache** that contains the forwarding information and a **MREQ-Cache** that stores recently received and processed MREQs. Protocol message formats and data structures used in GRAPE are presented in Appendix A and Appendix B respectively.

5.2.2 Construction of Cluster Structure

A two-tier non-overlapping stable cluster structure is essential to achieve efficient and reliable multicast routing in GRAPE. GRAPE can operate correctly with any existing cluster structure and the performance of GRAPE can be guaranteed if the cluster structure is stable throughout the network communication. Such a cluster structure can be easily formed by using Mobility-based D-Hop (hereafter known as MobDHop) clustering algorithm. MobDHop forms multihop clusters of flexible diameter, in which the diameter and cluster assignment are dependent on the mobility pattern in the networks. Thus, a stable cluster structure can be constructed when a network exhibits group mobility pattern. Moreover, the diameter of the clusters is not limited to any value. It is solely determined by the mobility characteristics in the networks. MobDHop identifies nodes which belong to the same group and then gathers these nodes into a stable cluster. Hence, GRAPE assumes the existence of MobDHop or similar clustering schemes to provide a stable cluster structure for its operation. In MobDHop, every cluster will be led by a clusterhead which is optimally located in the middle of the cluster. This property allows the clusterhead to exploit "wireless broadcast advantage" for multicast data dissemination within its cluster. Therefore, the dissemination of data packets from clusterhead to its members is done in an optimal fashion. The role of the clusterhead in GRAPE will be further elaborated in later sections.

5.2.3 Multicast Group Management Mechanism

Managing multicast groups is usually overlooked in previously proposed multicast routing protocols. These multicast routing protocols assume the source node or a specially elected core node/multicast group leader to maintain the membership of a multicast group. In GRAPE, the load of multicast group membership management is evenly distributed among source node and clusterheads in the network. Each clusterhead will be in charge of the group membership maintenance of its own cluster members. Therefore, group membership information is aggregated based on the cluster topology and effectively sent to the source node by each clusterhead. Figure 5.2 shows an example of the aggregation of multicast group membership information. As shown in Figure 5.2, only nodes 3 and 23 will join the multicast group on behalf of their cluster members. Source node (node 33 in this example) will only construct and maintain a multicast delivery structure to nodes 3 and 23.

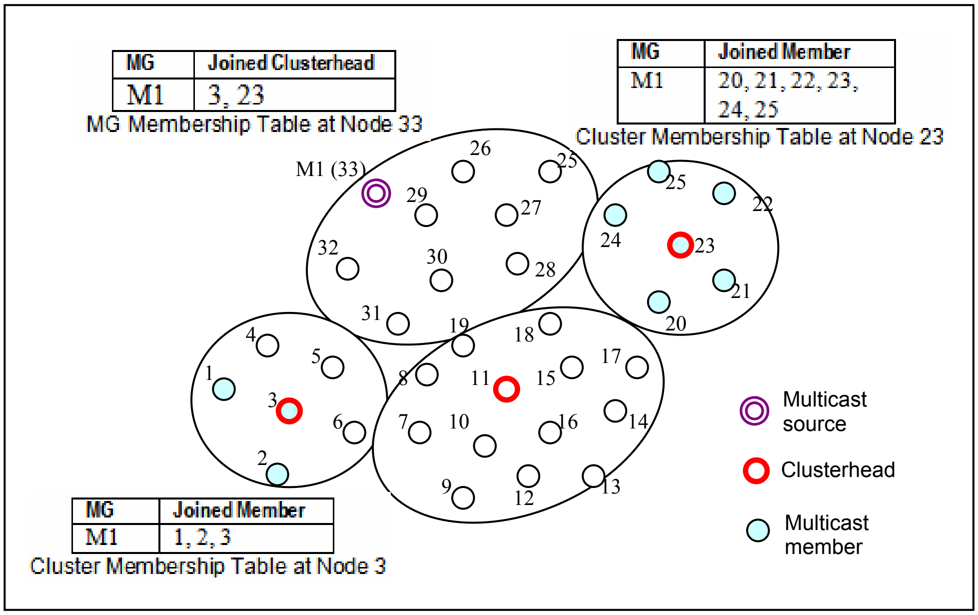


Figure 5.2 Aggregation of multicast group information.

5.2.3.1 Initiating a Multicast Group

A node becomes the multicast source node when it has data to send to a particular multicast group. This source node will first send a MREQ to construct the upper-tier multicast forwarding infrastructure. Upon receiving MREQ, clusterheads which are interested to join the multicast group session will reply with a MREP packet along the route where MREQ arrives. When the source node receives the MREP from clusterheads, it will update its MG-Membership-Table and start to forward data packets via the upper-tier multicast forwarding infrastructure constructed during the dissemination of MREQ.

5.2.3.2 Joining a Multicast Group

When a non-clusterhead (ordinary or gateway) node is interested to join a multicast group, it will send a MemberJoin packet to its clusterhead. Upon receiving MemberJoin from its cluster member, the clusterhead will update its Cluster-Membership-Table. When a clusterhead receives a MREQ from the source node, it will check its Cluster-Membership-Table. If there are members in its cluster that are interested to join this multicast group session, it will initiate a MREP packet back to the source node in order to join the multicast group on behalf of its cluster members. A rejoin operation will be initiated by an ordinary node when it joins a new cluster. When a node changes its clusterhead, it will resend MemberJoin packets to its new clusterhead based on the information related to its multicast group subscription.

5.2.3.3 Maintaining a Multicast Group

Group membership is refreshed by a periodical flooding of MREQ packet across the network by the source node. For instance, MREQ is generated at 20-seconds interval in all relevant simulations in this thesis, which is same as the join query refresh value in ODMRP for fair comparison. This is the most reliable method to ensure both membership and route freshness in the upper-tier multicast forwarding infrastructure. However, it will incur a substantial amount of overhead as the number of sources increases. To alleviate this problem, several methods such as probabilistic scheme, counter-based scheme, distance-based scheme, location-based scheme and cluster-based scheme as proposed in [88] may be adopted to reduce overhead during the flooding of MREQ.

5.2.3.4 Leaving a Multicast Group

Group members can leave a multicast group at anytime. A node which is associated to a clusterhead can send a MemberLeave packet to its clusterhead to indicate its intention to leave the multicast group session. Upon receiving MemberLeave packet, the clusterhead will remove this particular node from its Cluster-Membership-Table and check if the Cluster-Membership-Table is empty. If the Cluster-Membership-Table is empty, the clusterhead will implicitly leave the multicast group by not replying MREP in the next round of MREQ flooding.

5.2.4 Multicast Packet Forwarding Mechanism

In GRAPE, multicast packet forwarding is done in two levels. At the first level, a source-based multicast mesh is constructed and maintained. The mesh construction can be based on any general multicast path setup algorithm such as the most commonly used Shortest-Path heuristic. However, the shortest path heuristic may not construct an optimal multicast delivery infrastructure in terms of data overhead. Therefore, a new multicast path setup algorithm, namely Bandwidth-Optimized and Delay-Sensitive (BODS) algorithm, which constructs a more efficient multicast delivery infrastructure without sacrificing delay performance based on the Nearest-Participant heuristic, is proposed in this research. A detailed discussion and simulation results of this new multicast path setup algorithm will be presented in Chapter 6. However, a brief overview will be given here for the completeness of GRAPE discussion.

After the multicast mesh is constructed, multicast packets are then forwarded from the source node to every leaf node i.e. clusterhead that joins the multicast group via the mesh path. This is known as upper-tier multicast communication. At the second level, clusterheads that join the multicast group forward the multicast packets to those cluster members that are interested in the multicast communication. This is known as lower-tier multicast communication or intra-cluster forwarding. Two strategies are chosen dynamically by the clusterhead to efficiently forward multicast packets within its cluster, i.e. (a) cluster broadcasting or (b) stateless tree-based multicasting.

5.2.4.1 Upper-tier Multicast Communication

The upper tier multicast communication in GRAPE involves the dissemination of multicast packets from source node to clusterheads that join the multicast group. GRAPE constructs a source-based multicast forwarding mesh when a source node has data to send. Figure 5.3 depicts a flow chart that describes the process when a source node receives a data packet from the upper layer. When a source node receives a multicast data packet from the upper layer, it will first check for the availability of multicast forwarding mesh. If the multicast forwarding mesh has been constructed, the source node will forward the data packets according to the procedure as illustrated in Figure 5.4. When the intermediate node receives this data packet, it will check the MG-Flag-Cache to check if it is one of the forwarding nodes for this multicast group. If it is one of the forwarding nodes, it will rebroadcast the data packet accordingly. The process continues until the data packet reaches the intended multicast destinations. Mesh-based forwarding is chosen in this research since it is more suitable for the wireless networks and has been shown in [101] to be more effective and reliable. Most of the mesh-based protocols can achieve higher packet delivery ratio than their tree-based counterparts.

If the multicast forwarding mesh has not been constructed, the source node will initiate and broadcast a MREQ packet to build the forwarding mesh. The mesh construction is based on the BODS multicast path setup algorithm. The entire process of the construction of upper tier infrastructure is illustrated in Figure 5.5, Figure 5.6, and Figure 5.7.

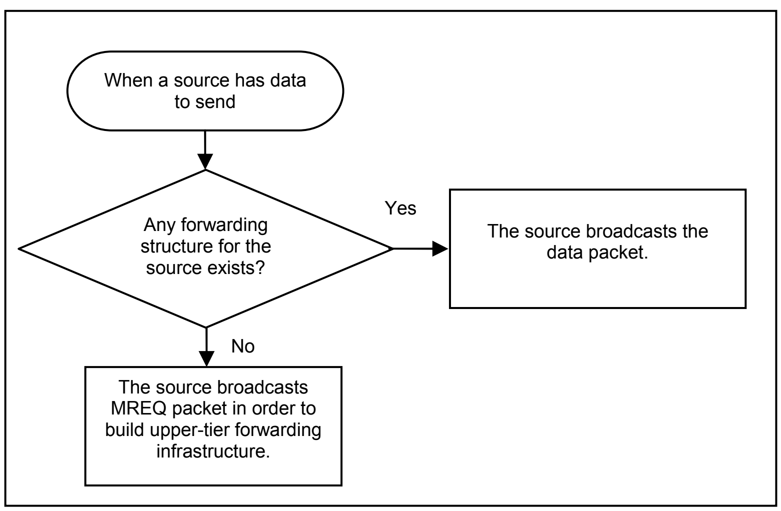


Figure 5.3 Flow chart for MREQ generation.

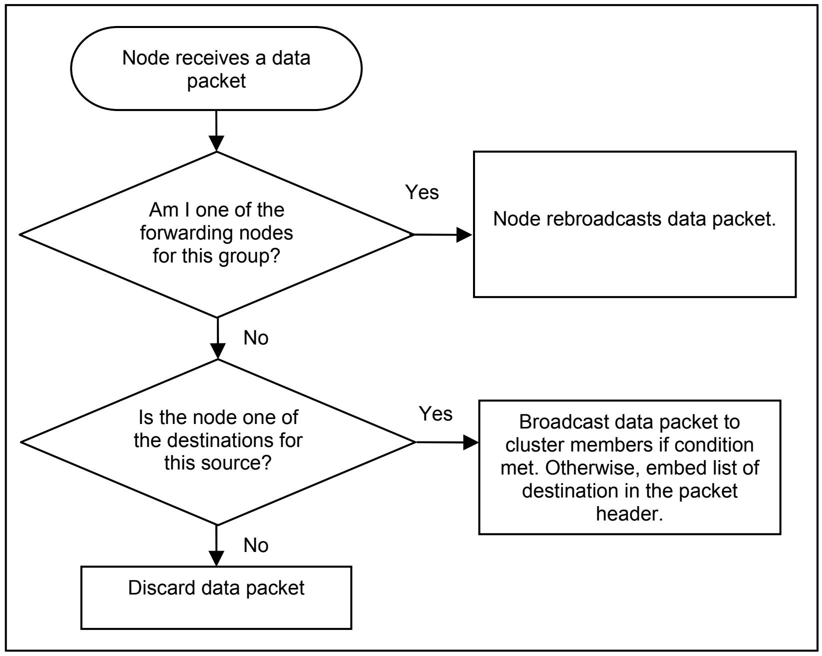


Figure 5.4 Flow chart for the forwarding of multicast data packets in GRAPE.

When a source node first receives a multicast data packet from the upper layer, it will initiate a MREQ packet transmission in the entire network. When an intermediate node first receives a MREQ packet, it will set a timer, MREQ-Delay-Timer, and wait for it to expire before it rebroadcasts the MREQ. The length of this delay timer is determined by the BODS algorithm. This is part of the BODS algorithm which aims to prioritize the selection of more optimal routes and thus form a more optimal multicast mesh. Before MREQ-Delay-Timer expires, all subsequent MREQs from the same source node will be stored in the MREQ-Cache. When MREQ-Delay-Timer expires, the intermediate node will process the MREQ-Cache based on BODS algorithm to determine two best routes as the primary and secondary path from the source node. The corresponding previous hops are selected as the primary and secondary previous hop respectively. This is the reverse path which will be used to forward MREP back to the source node.

The intermediate node will then forward MREQ with updated BODS information. Once the MREQ reaches the multicast destination, these destinations (interested clusterheads) respond by sending a MREP as broadcast packet back to the source node via selected primary and/or secondary previous hop. The secondary path serves two purposes here, i.e. (i) as the redundant path and (ii) as the back-up path. Since sending redundant packets helps to alleviate packet loss in highly mobile networks, a certain degree of path redundancy in MANETs is sometimes desirable. In most of the existing mesh-based multicast protocols such as ODMRP, path redundancy is not deterministic. Instead, the path redundancy cannot be predicted or imposed. In GRAPE, the path redundancy is deterministic and can be defined. A redundancy factor (ReF) with value zero to one is introduced in GRAPE to increase the level of data redundancy. When ReF equals zero, there is no path redundancy in the forwarding infrastructure. Only the primary path is used to forward data packets. When ReF is larger than zero, there is a probability equals to the value of ReF such that a node on the secondary path will be chosen as one of the forwarding nodes. When ReF equals one, GRAPE uses both primary and secondary paths to forward multicast data packets. Therefore, the data redundancy is higher. When the network mobility is high, high level of data redundancy can help to reduce packet loss. Furthermore, the secondary path can be used as the backup path when primary path failure is detected. This will help to avoid packet loss due to route failure. When a destination node (clusterhead) detects the disruption in the arrival of data packets, it will initiate a MREP back to the source node via the secondary path. This MREP will be routed back to the source node via the secondary path selected based on the BODS algorithm. In this case, all nodes along the secondary path will become members of the forwarding mesh.

Upon receiving the MREP, an intermediate node will set the MG-FLAG true if it finds that its address is stated as primary previous hop in MREP. If the address of the intermediate node is stated as secondary previous hop, the intermediate node will set the MG-FLAG true with certain probability (i.e. probability = ReF). The intermediate node will then continue to forward MREP back to source node by updating the selected primary and secondary previous hop. Upon receiving the first MREP, the multicast source will begin the multicast packet forwarding.

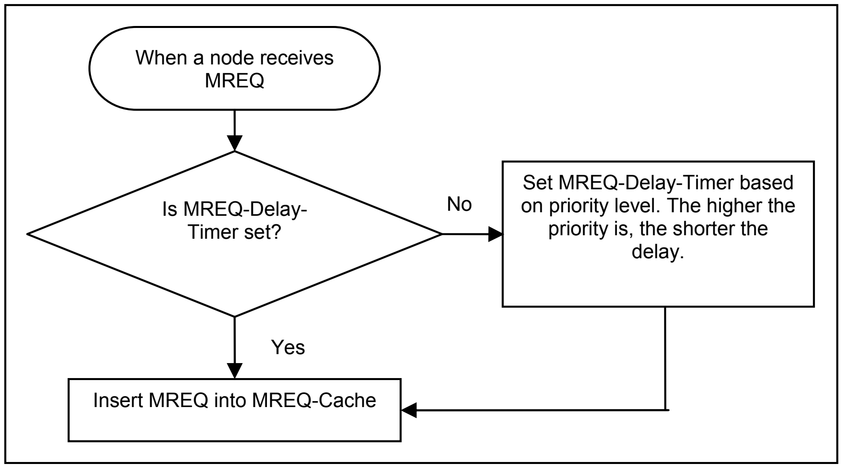


Figure 5.5 Flow chart for MREQ handling at every node.

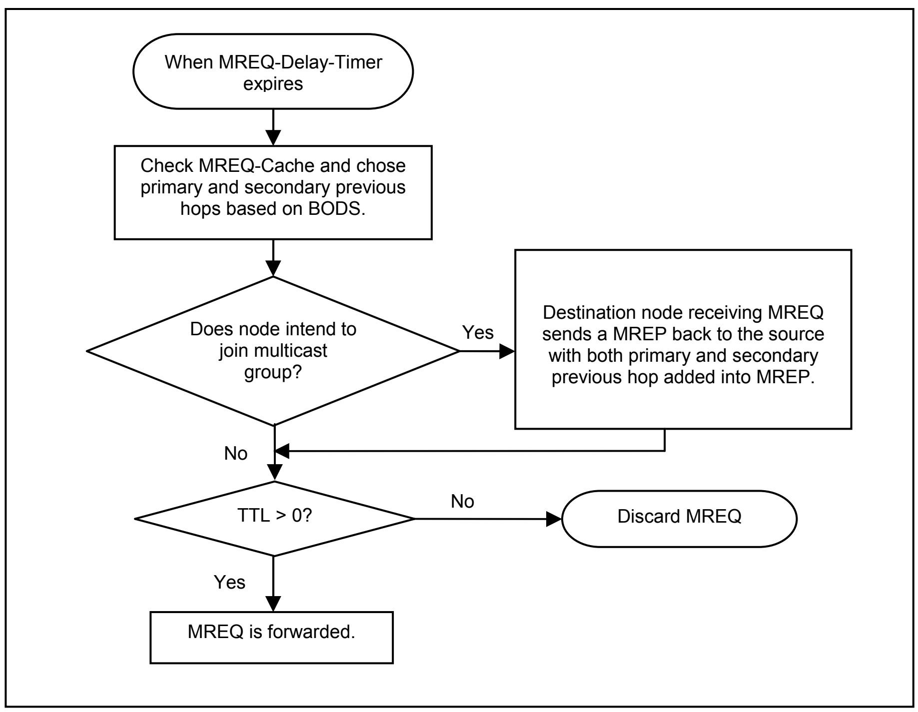


Figure 5.6 Flow chart for the construction of upper-tier multicast delivery tree.

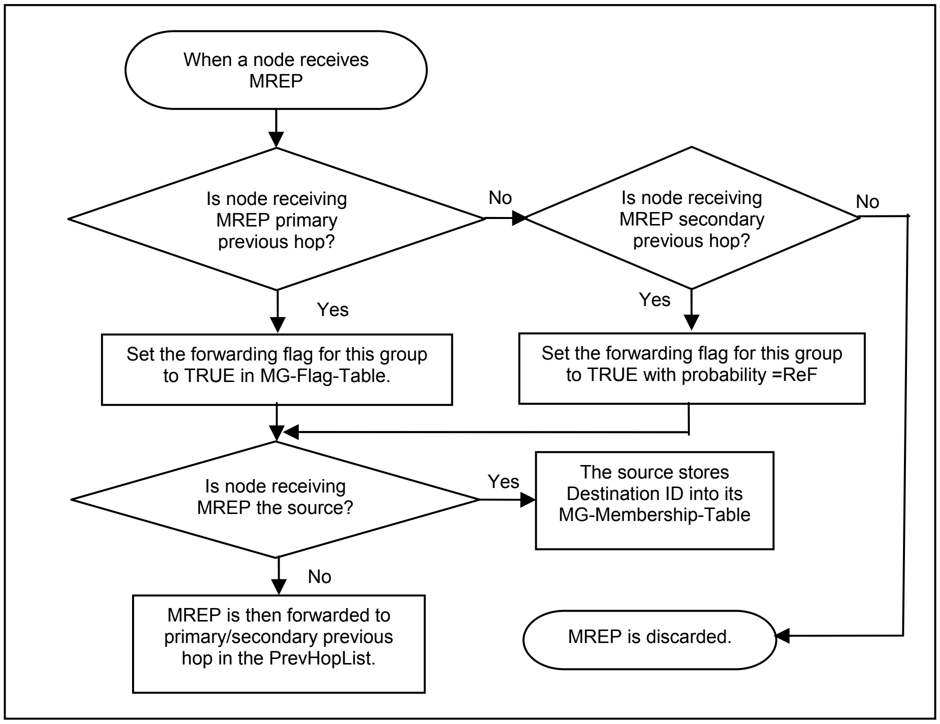


Figure 5.7 Flow chart for the MREP handling in GRAPE.

Due to the dynamism of network topology, the upper-tier multicast structure should be consistently maintained and updated throughout the entire multicast session. In GRAPE, the source refreshes the source-based multicast mesh by sending MREQ periodically at a predefined interval. This mechanism is similar to the one used in ODMRP. By doing this, the multicast routing information can be updated and the source-based multicast mesh might also be refined to a more optimal mesh during the refresh process.

5.2.4.2 Lower-tier Multicast Communication

The lower-tier multicast communication in GRAPE involves the dissemination of multicast packets from clusterheads to their members that join the multicast group. Based on the multicast traffic characteristics within their clusters, each clusterhead will choose an optimal forwarding strategy. It is observed in [141] that a simple broadcast scheme can significantly reduce the control overhead in scenario wherein the density of group members is high. The authors suggested that broadcasting is more efficient when 40% or more of the nodes in the network are multicast group members. Based on this observation, the clusterhead in GRAPE will choose to broadcast multicast packets to its member when more than 40% of the total cluster members join the relevant multicast group in order to fully exploit the advantage of wireless medium.

If the multicast packets are aimed for relatively fewer members (less or equal to 40% of the total cluster members), clusterhead will choose to forward multicast packets to its member by encapsulating the address list of these members into the header of each multicast packet like the forwarding mechanism used by Differential Destination Multicast (DDM). There is no explicit multicast tree to be maintained within the clusters. DDM-like multicasting is more efficient in a relatively larger cluster where only a small portion of the cluster members join the same multicast group. It is assumed that each node will maintain a list of all its descendant nodes for the correct operation of DDM-like packet forwarding. Upon receiving a data packet, a node will send the packet to upper layer application if it is one of the multicast receivers. Then it will check the packet header for possible address list. If the address list is not empty and there exists addresses which are in its descendant list, it will forward the packet to its child nodes. The address list in the header of the data packet will be truncated by including only the addresses of its descendant nodes. If the address list is empty, the packet will be discarded. The process continues until the packet reaches the boundary of the cluster.

The maintenance of the lower-tier communication is the responsibility of the clusterhead, which is elected based on the underlying clustering algorithm used. It is clear that a stable cluster structure is important in GRAPE since the clusterhead changes will inevitably cause all its cluster members to rejoin the multicast session via newly elected clusterhead. Packet loss may happen during these changes. Stable cluster structure is usually one of the important goals a MANET clustering algorithm tries to achieve. For example, MobDHop has been shown to form and maintain stable multihop clusters in this thesis.

5.3 Summary

In this chapter, GRAPE multicast routing protocol, a hierarchical multicast routing protocol that works on top of a stable cluster structure, has been introduced. It proposes a new multicast group management scheme that distributes the management load to all clusterheads in the network. Apart from this, GRAPE also introduces two-tier multicast routing that adapts to different traffic properties. The upper-tier multicast communication structure connects source to clusterheads that are interested to join the multicast communication on behalf of their members. The packets dissemination for upper-tier structure is done in a more efficient Steiner-like mesh which is constructed by a new multicast algorithm, Bandwidth-Optimized and Delay-Sensitive (BODS) algorithm which will be further elaborated in the following chapter.

The lower-tier multicast communication infrastructure connects clusterheads and its members. Clusterheads dynamically select a suitable forwarding scheme, i.e. either cluster broadcasting or stateless tree-based multicasting to forward packets to its members based on the traffic characteristic within their clusters. It may switch from one scheme to another if the traffic within cluster changes. The robustness of the protocol is further enhanced by introducing the multi-path property which is widely used in unicast routing. A redundancy factor is introduced in order to provide deterministic path redundancy in GRAPE. In short, GRAPE offers a scalable, flexible, adaptive multi-path multicast routing solution for MANETs that is suitable for various kinds of network configuration and applications. The effectiveness and benefits of GRAPE will be evaluated via simulation approach in Chapter 7.

CHAPTER 6

BANDWIDTH-OPTIMIZED AND DELAY-SENSITIVE MULTICAST PATH SETUP ALGORITHM

6.1 Introduction

Control overhead has been considered as an important metric in the evaluation of a multicast routing protocol in MANETs. However, it is equally, if not more, important to consider the amount of overhead incurred by sending unnecessary duplicate data packets since these packets usually consume more bandwidth. Furthermore, sending unnecessary data packets may cause more MAC layer contentions and collisions in IEEE 802.11b wireless networks where broadcast/multicast data is sent blindly without collision avoidance mechanism. Most existing MANET multicast routing protocols build shortest-path trees/meshes or sub-optimal shared trees/meshes instead of bandwidth-optimal (highest forwarding efficiency) multicast structure. Computing the bandwidth-optimal multicast structure is also known as the minimum Steiner tree problem in graph theory which is known to be NP-complete [81]. Due to the fact that building minimum Steiner tree is computationally expensive and almost infeasible in resource-scarce MANETs, there has been little work done in this area. Hence, it is necessary to investigate and propose an optimal multicast algorithm by taking salient MANET characteristics into consideration. In this thesis, a distributed multicast path setup algorithm, which constructs an efficient multicast delivery structure based on Nearest-Participant heuristic in order to reduce the number of forwarding nodes and hence the number of redundant packets (data overhead) as well as possible collisions, is proposed.

Here, we are interested to find a more optimal multicast forwarding infrastructure in MANETs in terms of bandwidth consumption, considering the salient characteristics such as decentralized control, constrained bandwidth, and the absence of the information on global network topology. Therefore, the problem has been reformulated as the construction of a Steiner-like forwarding structure based on the partial topology information available in a distributed and online manner. In the following sections, a distributed multicast path setup algorithm, which constructs a multicast forwarding structure that considers only vertex addition at every incremental step, is presented. The main objective of this algorithm is to construct bandwidth-optimal multicast tree in order to minimize packet redundancy as well as the possibility of collisions. However, such a delivery tree may incur higher delay since the path length between pairs of source and destination may not be the shortest. Thus, the proposed algorithm attempts to construct a hybrid of bandwidth-optimal Steiner tree and shortest path tree in order to build an efficient multicast delivery infrastructure without sacrificing the delay performance. This algorithm is named as Bandwidth-Optimized and Delay-Sensitive (BODS) multicast path setup algorithm and it is suitable for applications that are both bandwidth intensive and delay sensitive such as multimedia streaming applications. In section 6.2, the problem statement is formulated. This is followed by the assumptions on the network model. The proposed algorithm, BODS multicast path setup algorithm, is described in section 6.3. Simulations have been conducted to evaluate the multicast efficiency and the delay performance of BODS algorithm. Simulation results and discussions are presented in section 6.4.

6.2 Network Model and Problem Formulation

A MANET is represented by an undirected graph $G(V,E)$ where V is the set of vertices and E is the set of edges. The network is assumed to be two dimensional and mobile nodes are represented by vertices of the graphs. Each node $v \in V$ has a transmission range of r. Let $d(v_1,v_2)$ be the distance between two vertices v_1, $v_2 \in V$. An edge between two vertices v_1 and v_2 exists if and only if $d(v_1,v_2) \leq r$. It is assumed that all links are bidirectional, i.e. $(v_1,v_2) \in E \Leftrightarrow (v_2,v_1) \in E$. We are given G with a cost function, $C:E \rightarrow \boldsymbol{R}$, and a source node s.

In the source-initiated multicast path setup of an ad hoc network, multicast receivers $R = \{v_1, v_2, \ldots, v_k\}$ arrive in an online but ordered fashion. The closer multicast receivers are usually revealed before those that are located further away from the source node. Our problem is, therefore, to construct a tree T connecting s to all the receivers that have been revealed so far without the full knowledge of already constructed tree. Since each multicast receiver must choose a path and send a reply packet to join the multicast group upon receiving the request packet via the chosen path, it is impossible for the reacting node to be completely aware of the already built multicast delivery tree. Hence, the construction of T should be conducted in a fully distributed manner since each node is only equipped with partial, most probably local topology information. Let $d_G(s,v)$ be the shortest path distance from s to v in the network, $d_T(s,v)$ be the path length from s to v in T and $d_K(s,v)$ be the known shortest path distance from s to v.

6.3 BODS Multicast Path Setup Algorithm

The BODS multicast path setup algorithm forms a multicast mesh in a distributed manner due to the salient characteristics of MANETs. In the next sub-section, we describe the details of BODS algorithm which aims to construct a source-based multicast mesh of low-cost (low data overhead) and good delay. We also discuss the integration of BODS algorithm into ODMRP in the following section.

6.3.1 Nearest-Participant Heuristic

The input to BODS is a series of "request to join" to multicast groups by interested multicast receivers. The source node will first broadcast a query packet with two extra fields, i.e. Nearest-Participant, v_p, and Distance to Nearest-Participant, $d(v_p,v_i)$. Upon receiving this query packet, node i will check the Nearest-Participant field to determine the priority of this query packet based on Table 6.1 before forwarding this query packet. If this field is not empty, this implies that the query packet has arrived from a path that consists of other multicast destinations or forwarding nodes as intermediate nodes which have already joined the multicast group before node i. Therefore, these packets should be given higher priority. Otherwise, this query packet will be given a lower priority.

After deciding the priority of this query packet, node i will trigger a delay timer based on the priority chosen. The higher the priority is, the shorter the delay. The shorter the delay, the sooner the packet will be rebroadcast to other nodes. Therefore, the query packet with a higher priority should arrive at other multicast destinations which are further away from the source node slightly earlier than other query packets. Hence, these paths are prioritized over other paths that do not consist of multicast destinations or existing forwarding nodes as intermediate nodes. There are two purposes of setting a delay timer: (a) to accumulate knowledge about other paths and (2) to avoid long paths. Since the timer will be triggered when the first query packet is received and expire after a certain amount of time, a path to node i that incurs large delay will not be considered in the path selection. Before the delay timer expires, node i will continue to collect query packets that arrive via other paths. When this timer expires, node i will make a decision on which path to choose and forward the query packet accordingly. If more than one path is known, node i will choose the best path (with minimum hop count from source) with non-empty Nearest-Participant field if the distance to the nearest participant is no larger than the distance of the known shortest path from s multiplied by a factor, β, which is in the range of 0 and 1, as shown in Eq. 6.1 and the length of this path is also less than or equal to two times the known shortest path length as shown in Eq. 6.2.

$$d(v_p, v_i) < \beta \bullet d_K(s, v_i) \qquad \text{(Eq. 6.1)}$$

$$d(s, v_i) \leq 2 \bullet d_K(s, v_i) \qquad \text{(Eq. 6.2)}$$

If such a path does not exist, node *i* will choose the shortest path. Different values of the β factor, allow the construction of a shortest path tree, or a combination of a greedy Steiner tree and shortest path tree. When β equals 0, a shortest path tree will be formed. Otherwise, a hybrid of a greedy Steiner tree and shortest path tree is constructed. When β equals 1, the path length from the source to each destination can be guaranteed to be at most two times the length of the shortest path between the source and destination (as shown in Eq. 6.3) if the delay value of different priority level is carefully chosen.

$$d_T(s, v_i) \leq 2 \bullet d_G(s, v_i) \qquad \text{(Eq. 6.3)}$$

The selection of the delay timer value for different priority level will be discussed in the following section. If node *i* is one of the multicast destinations or one of the forwarding nodes in the existing multicast mesh, it will add its address into the Nearest-Participant field and reset the Distance to Nearest-Participant field to zero. After the decision is made, node *i* will forward the query packet. The query packet will be propagated to the entire connected component within the network to allow all multicast destinations to join the multicast mesh for each path setup process. Figure 6.1 and Figure 6.2 illustrate the operation of BODS algorithm upon receiving query packet (MREQ) from source node and upon the expiration of delay timer respectively in *pseudocode*.

Table 6.1 Priority level used in BODS algorithm

Priority Level	Condition
Highest	Reacting node is a multicast destination and this MREQ arrives via a path that contains other multicast destinations or existing forwarding nodes as intermediate nodes.
Intermediate	Reacting node is a multicast destination and this MREQ arrives via a path that does not contain other multicast destinations or existing forwarding nodes as intermediate nodes.
	Reacting node is not a multicast destination, but this MREQ arrives via a path that contains other multicast destinations or existing forwarding nodes as intermediate nodes.
Lowest	Reacting node is not a multicast destination, and this MREQ arrives via a path that does not contain other multicast destinations or existing forwarding nodes as intermediate nodes.

```
BEGIN
IF (Lookup_MREQCache(MREQ->seq_no) = TRUE) THEN
        IF (DelayTimerExpired() = FALSE) THEN
                Insert_MREQCache(MREQ)
        ELSE
                Discard_Packet(MREQ)
        END IF
ElSE
        IF (Lookup_MREQCache(MREQ->seq_no) = FALSE) THEN
                IF ((MREQ->NP <> NULL) &&
                    (MulticastReceiver(v) = TRUE)) THEN
                        DelayTimerValue = HIGH_PRIORITY_VALUE
                ELSE
                        IF (((MREQ->NP = NULL) &&
                            (MulticastReceiver(v) = TRUE)) ||
                           ((MREQ->NP <> NULL) &&
                         (MulticastRecevier(v) = FALSE))) THEN
                                DelayTimerValue = INTERMEDIATE_PRIORITY_VALUE
                        END IF
                ELSE
                        IF ((MREQ->NP = NULL) &&
                           (MulticastReceiver(v) = FALSE)) THEN
                                DelayTimerValue = LOW_PRIORITY_VALUE
                        END IF
                END IF
        END IF
        IF (MREQ->ttl > 0) THEN
                SetDelayTimer(DelayTimerValue)
        ELSE
                Discard_Packet(MREQ)
        END IF
END IF
END
```

Figure 6.1 **Pseudocode upon receiving multicast route query packet (MREQ).**

```
BEGIN
MREQEntry = StartOf(MREQCache)
REPEAT
        IF (MREQEntry->NP = NULL) THEN
                IF (MREQEntry->SPDist < CurrShortestDist)
                        CurrShortestDist = MREQEntry->SPDist
                        CurrLowestDistToNP = 0
                        CurrPrevHopIPAddr = MREQEntry->PrevHopIPAddr
                END IF
        END IF
        NextOf(MREQCache)
UNTIL (EndOf(MREQCache))

MREQEntry = StartOf(MREQCache)
REPEAT
        IF (MREQEntry->NP <> NULL) THEN
                IF ((MREQEntry->DistToNP < CurrLowestDistToNP) &&
                    (MREQEntry->DistToNP < β*CurrShortestDist) &&
                    (MREQEntry->CurrFwdCount <= 2*CurrShortestDist)) THEN
                        CurrNP = MREQEntry->NP
                        CurrLowestDistToNP = MREQEntry->DistToNP
                        CurrPrevHopIPAddr = MREQEntry->PrevHopIPAddr
                        CurrFwdCount = MREQEntry->FwdCount
                END IF
        END IF
        NextOf(MREQCache)
UNTIL (EndOf(MREQCache))

IF (MREQCache->ttl > 0) THEN
        MREQPacket = Allocate_Packet(MREQEntry)
        MREQPacket->FwdCount = CurrFwdCount + 1
        IF (MulticastReceiver(v) = TRUE) THEN
                MREQPacket->NP = v
                MREQPacket->DistToNP = 0
        ELSE
                MREQPacket->NP = CurrNP
                MREQPacket->DistToNP = CurrLowestDistToNP
        END IF
        Forward_Packet(MREQPacket)
END IF
END
```

Figure 6.2 Pseudocode upon the expiration of delay timer.

6.3.2 Selection of Delay Value

We assume that σ is the average per-hop-delay in the network. Let L be the distance of the shortest path between <source, destination> pair which is the largest in the network, $l \times \sigma$ be the delay value for the lowest priority packet, and $h \times \sigma$ be the delay value for the highest priority packet. To ensure that the path length from the source node to the destination to be at most two times the length of the known shortest path between <source, destination> pair when β equals 1, we must compute a suitable value of h based on the value of l and L. The value of l must be carefully chosen since it will affect the total setup time of the multicast forwarding structure. If l is too large, the setup time of the multicast mesh may be too long. However, having a small l might reduce the tree to a shortest path tree since the possibility of getting different Join-Querys arriving from different path is smaller.

Figure 6.3 shows the worst case when Join-Query-1 arrives at node R via a path in which all intermediate nodes is multicast receiver. In this figure, the value shown in the rectangle beside each node indicates the arrival time of the related packet. Join-Query-1, in this case, will be prioritized along the way and might reach node R earlier than Join-Query-2 if the delay value is not carefully chosen. We need to ensure that the Join-Query-2 arrives at node R earlier than Join-Query-1 or Join-Query-2 arrives at node R before the delay timer at node R expires in order to avoid choosing path which is two times longer than the shortest path. If Join-Query-1 is the first query packet that arrives at node R, the delay timer will be triggered. In this case, Join-Query-2 must arrive at node R before its delay timer expires. Therefore, $2\sigma + l \cdot \sigma$ must be less than $5 \cdot \sigma + 4 \cdot h \cdot \sigma + h \cdot \sigma$. The last term ($h \cdot \sigma$) corresponds to the length of delay timer which will be set if Join-Query-1 arrives at node R before Join-Query-2. We generalize this situation by using L, l and h. Solving the following equation will give a suitable value of h:

$$(L-1) \cdot l \cdot \sigma + L \cdot \sigma \leq (2L+1) \cdot \sigma + 2 \cdot L \cdot h \cdot \sigma + h \cdot \sigma \qquad \text{(Eq. 6.4)}$$

The suitable value of h is therefore given by the following equation:

$$h \geq \frac{(l+1) \cdot L - l}{2L+1} - 1 \qquad \text{(Eq. 6.5)}$$

For example, given $L = 8$ and $l = 10$, h must be at least 3.58. By choosing h equals 4, the algorithm sets the length of delay timer for the highest priority packet to be 4σ while the length of delay timer for the lowest priority packet to be 10σ. For intermediate priority packet, a suitable value for i can be chosen as long as the following criteria fulfils: $l<i<h$. In our simulation setup, i equals $(l+h)/2$.

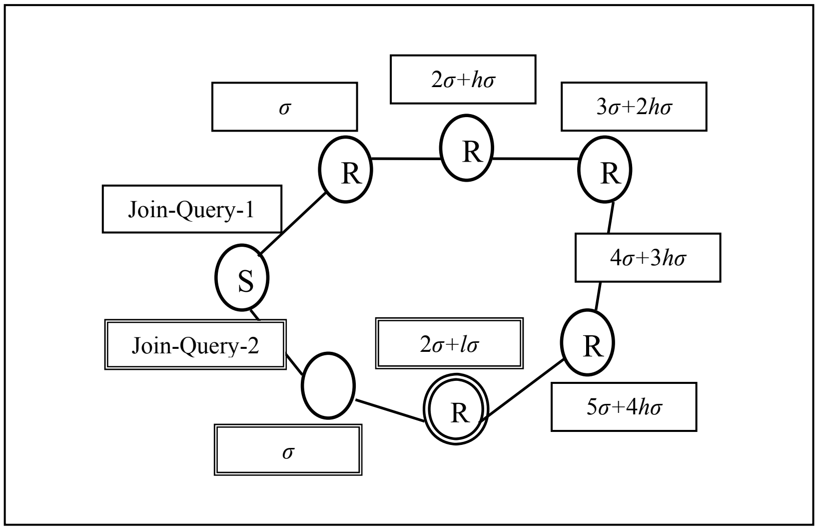


Figure 6.3 **Determination of the value of *l*.**

6.3.3 Illustration by Example

In this example, assume that the highest, intermediate and lowest priority delay value is set to 3, 6 and 9 milliseconds respectively. As illustrated in Figure 6.4, source 1 first broadcasts a MREQ packet with both Nearest-Participant field and Hop-Count-to-Nearest-Participant (HCNP) field set to NULL. Node 3 and node 2 receive MREQ from the source node and set their timer according to the corresponding priority. Since both node 3 and 2 are not multicast destination and both Nearest-Participant field in MREQ are empty, the lowest priority is chosen. Thus, a timer of 9 milliseconds is set. When this timer expires, both node 3 and 2 broadcast MREQ. Node 4, which is one of the multicast destinations, receives MREQ and determines the priority as intermediate level. Therefore, a timer of 6 milliseconds is set at node 4. When timer expires, node 4 will choose the path via previous hop, i.e. node 3. Before re-broadcasting MREQ, node 4 updates the Nearest-Participant field with its own address and sets HCNP field to 0. Node 7, another multicast destination receives the first MREQ via node 5 and starts its 6-milliseconds-timer (intermediate priority). During this period, node 7 receives another MREQ via node 6 which consists of an intermediate multicast destination i.e. node 4. Since the hop count to node 4 from node 7 equals 2 and is less than the known shortest path length, i.e. 3, node 7 decides to choose node 6 as the previous hop (and node

5 as secondary previous hop as required by GRAPE protocol). The process continues until MREQ traverses the entire network. It is obvious in this example that the number of forwarding nodes in a shortest-path tree and optimal Steiner tree is the same (i.e. 4). However, it is important to note that by using the optimal tree, we avoid choosing both node 2 and node 3 as forwarding nodes. This minimizes the probability of having MAC contention and collisions between node 2 and node 3.

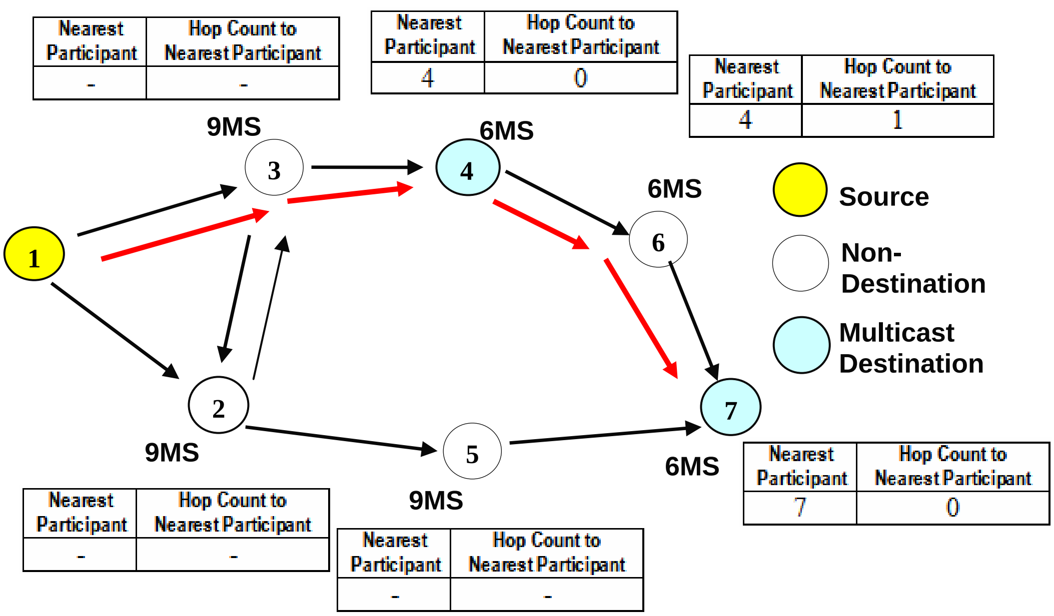


Figure 6.4 **The operation of BODS multicast path setup algorithm.**

6.3.4 Integration of BODS into ODMRP

In general, the BODS algorithm can be used as the underlying multicast path setup algorithm for any multicast routing protocol. ODMRP is chosen in in order to demonstrate the performance of BODS since ODMRP has been shown to outperform other multicast protocols in its class [101]. The integration of BODS into ODMRP is simple and straightforward. Two new fields, i.e. Nearest-Participant and Hop-Count-to-Nearest-Participant (HCNP), are added into the header of every join query message propagated by the ODMRP source node.

On the other hand, the route selection process in ODMRP needs to be modified as explained in Section 6.3.1. Upon receiving a new join query packet, each node will determine its priority based on the information in Nearest-Participant field and set the delay timer accordingly. When the delay timer expires, the join query packet will be rebroadcast and the selection of route will be finalized based on the path knowledge accumulated before the timer expires, following the rules as narrated in previous sections.

6.4 Simulation Results and Discussions

In order to evaluate the performance of BODS, we implemented the integration of BODS into ODMRP protocol in QualNet 3.8. The performance of ODMRP with BODS was compared against the performance of original ODMRP under similar network configurations. The communication range was 376 metres which is the default value for IEEE 802.11 DCF with channel capacity of 2Mbps in QualNet simulator. For each network configuration, ten different scenarios were randomly generated using different seed and the average value of collected data was presented.

In the first set of simulations, nodes moved according to the Random Waypoint (RW) mobility model at the maximum speed of 2m/s and zero pause time. The duration of each simulation was 900 seconds. Each multicast source starts to generate constant bit rate (CBR) traffic at the rate of four 512-bytes data packets per second one after another (with the starting instances separated by 2 seconds) for 600 seconds. To evaluate the performance of BODS with respect to increasing multicast group size, the number of receivers was varied from 5 to 30 in the first set of RW simulations (RW-1). In this scenario, 400 nodes were simulated over an area of 2500 metres x 2500 metres. At the same time the performance of BODS algorithm was also tested by varying the number of source node in a group from 1 to 3. In the second set of RW simulations (RW-2), five scenarios: 144 by 1, 72 by 2, 48 by 3, 36 by 4 and 24 by 6, were tested to evaluate the performance of BODS with respect to the increasing number of active multicast sessions. Here

"72 by 2" means the scenario consisted of two multicast groups and 72 members per multicast group. Hence, in all scenarios, there were 144 multicast receivers in total. There was one source for each multicast group. The traffic demand remained the same in all scenarios.

In the second set of simulations, nodes moved according to the Reference Point Group Model (RPGM) at the maximum speed of 2 m/s and zero pause time. Each mobility group consists of ten nodes. The multicast application layer sources in this scenario generated CBR traffic at 2 packets per second. To evaluate the performance of BODS with respect to increasing multicast group size, the number of multicast receivers was varied from 20 to 100. The number of source node per group was 4 and 5 respectively.

Table 6.2 Simulation parameters

Parameter	Random Waypoint Model		Reference Point Group Model
	RW-1	RW-2	RPGM-1
Node Number	400	144	400
Number of Receivers	5, 10, 15, 20, 30	144, 72, 48, 36, 24	20, 40, 60, 80, 100
Number of Multicast Groups	1	1, 2, 3, 4, 6	1
Number of Source node per group	1, 2, 3	1	4, 5
Packets per second	4	4	2
Packet size (byte)	512	512	512
Simulation Duration (s)	900	900	900
Simulation Area (m^2)	2500x2500	2500x2500	2500x2500
Max speed (m/s)	2	2	2
Number of members per mobility group	-	-	10
Simulation Purpose	Scalability against multicast group size	Scalability against number of multicast groups	Performance under group mobility model

6.4.1 ODMRP and BODS Parameters

Protocol parameters and their corresponding values used in the simulations of ODMRP and BODS are listed in Table 6.3. The protocol parameters of ODMRP conformed to the default values suggested in the ODMRP Internet Draft version 4. For BODS, β was set to 1 and the corresponding delay values for different priority levels were computed as suggested in Section 6.4.2.

Table 6.3 ODMRP and BODS parameters

Parameter	Value
ODMRP Refresh Interval	20 sec
ODMRP FG_FLAG Timeout	60 sec
ODMRP Maximum Retransmission of Join Reply	3
ODMRP ACK for Join Reply Timeout	0.075 sec
ODMRP Aggregation of Join Reply Interval	0.025 sec
BODS β	1.0
Delay for highest priority Join_Query	4.0 msec
Delay for lowest priority Join_Query	10.0 msec
Delay for intermediate priority Join_Query	7.0 msec

6.4.2 Performance Metrics

The following performance metrics, which are similar to the set of performance metrics used in protocol evaluation in [101][115][122], were used in the performance evaluation of the effectiveness of BODS algorithm:

i. *Packet Delivery Ratio* (PDR): The number of data packet successfully delivered to multicast destinations over the number of data packets to be delivered to multicast destinations. The PDR value of “1” means all packets are successfully delivered to all multicast receivers.

ii. *Normalized Data Overhead*: The total number of data packets transmitted by both source node and intermediate nodes over the total number of data packets successfully delivered to multicast destinations. A larger value indicates that the protocol incurs higher data overhead and thus less efficient.

iii. *Normalized Control Overhead*: The total number of control packets transmitted over the total number of data packets successfully delivered to multicast destinations. A larger value indicates that the protocol incurs higher control overhead.

iv. *Mean Delivery Latency*: The mean difference between the time at which a data packet is generated and the time at which it is received by the multicast destinations. The mean latency is computed independently for each receiver and then the values are averaged across all multicast receivers.

6.4.3 Evaluation based on Random Waypoint Mobility

Figure 6.5 and Figure 6.6 show the performance of both ODMRP with BODS and the original ODMRP as functions of group size, i.e. the number of multicast receivers in the one-source and two-source RW scenario respectively. As shown in Figure 6.5(a) and Figure 6.6(a), both variants delivered more than 95% of the traffic in both one-source and two-source scenario. In most scenarios, the performance of both variants in terms of PDR was comparable. However, ODMRP incurred about 15%-30% more data packet transmissions than the proposed variant which was enhanced by BODS multicast path setup algorithm in order to achieve high PDR in scenarios which consist of 5 to 20 receivers. The additional data overhead may cause lower throughput in networks should the network load is increased. As shown in both Figure 6.5(b) and Figure 6.6(b), the differences in forwarding efficiency (data overhead) between these two variants became smaller when the number of multicast receivers was increased. This is because as the number of multicast receiver increased, the number of forwarding nodes that are needed to ensure connectedness of mesh also increased. Therefore, the gain of BODS algorithm over the shortest path algorithm became less significant. However, BODS still cut down 15% of data overhead in the 30-receiver scenario. The amount of control overhead generated by both variants was similar since BODS does not introduce additional control packets to the original ODMRP protocol.

Though ODMRP is expected to offer lower latency since it is using the shortest path algorithm, the performance of our proposed alternative outperformed ODMRP by about 10% as depicted in both Figure 6.5(d) and Figure 6.6(d). This is mainly attributed to the nature of BODS which is delay sensitive and also the reduction in data overhead. Reducing the amount of data that needs to be sent over the network relieves MAC-layer contentions and reduces collisions among multicast frames. Thus the latency between source and receivers can be shortened and the packet loss can be reduced.

Figure 6.7 shows the performance results for scenarios with three source nodes in RW simulations. The traffic load in these scenarios was much higher and the network was more congested. This is shown by the reduction in the PDR and much longer delivery latency. As shown in Figure 6.7(a), the PDR of the original ODMRP dropped below 95% when the number of multicast receivers was increased to 20 and 30 respectively. This is because the original ODMRP incurred a large amount of redundant data transmissions by choosing a large set of forwarding nodes. This created a substantial amount of contention and collision at the MAC layer while trying to broadcast unnecessary data packets. Meanwhile, ODMRP with BODS successfully delivered more than 95% of data packets in all cases. A slight drop in PDR was observed when the number of receivers was increased to 20 and 30. The original ODMRP incurs about 30% more data overhead than ODMRP with BODS in most cases as shown in Figure 6.7(b). The mean delivery latency for the original ODMRP increased drastically with the number of receivers in the network as shown in Figure 6.7(d). Meanwhile, ODMRP with BODS managed to deliver packet within 100 milliseconds except for the case where the number of multicast receivers was increased to 30.

Figure 6.8, on the other hand, shows the simulation results of scenario RW-2 where the number of active multicast sessions was varied from 1 to 6. The performance of both ODMRP with BODS and the original ODMRP was comparable where both protocols achieved similar PDR as shown in Figure 6.8(a). However, BODS is able to enhance the performance of ODMRP by incurring less data overhead and reducing mean delivery latency as shown in Figure 6.8(b) and Figure 6.8(d).

6.4.4 Evaluation based on RPGM

Figure 6.9 and Figure 6.10 depict the performance of both ODMRP with BODS and the original ODMRP as functions of active group for four-source and five-source RPGM scenario. As shown in Figure 6.9(a) and Figure 6.10(a), ODMRP with BODS consistently achieved slightly higher PDR than the original ODMRP and reduced the data overhead by 10 to 20%. BODS also improved the performance of ODMRP in terms of mean delivery latency especially in the scenario where the high traffic load was imposed. This again can be attributed to the nature of BODS which is delay sensitive and also the more optimal multicast delivery structure that leads to a reduction in data overhead which, in turn, reduces MAC contentions and collisions.

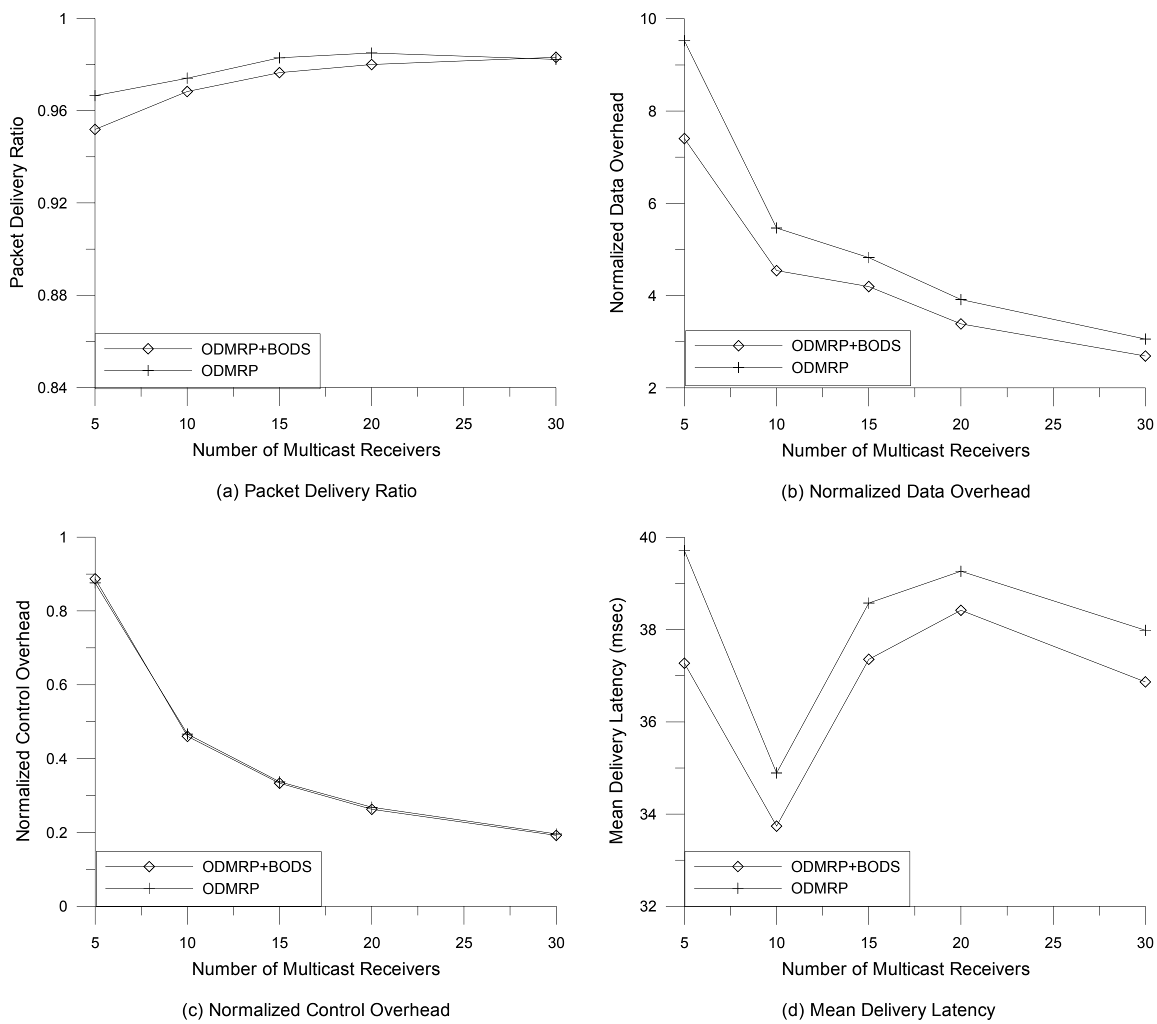


Figure 6.5 Performance versus number of multicast receivers in one-source scenario under RW model.

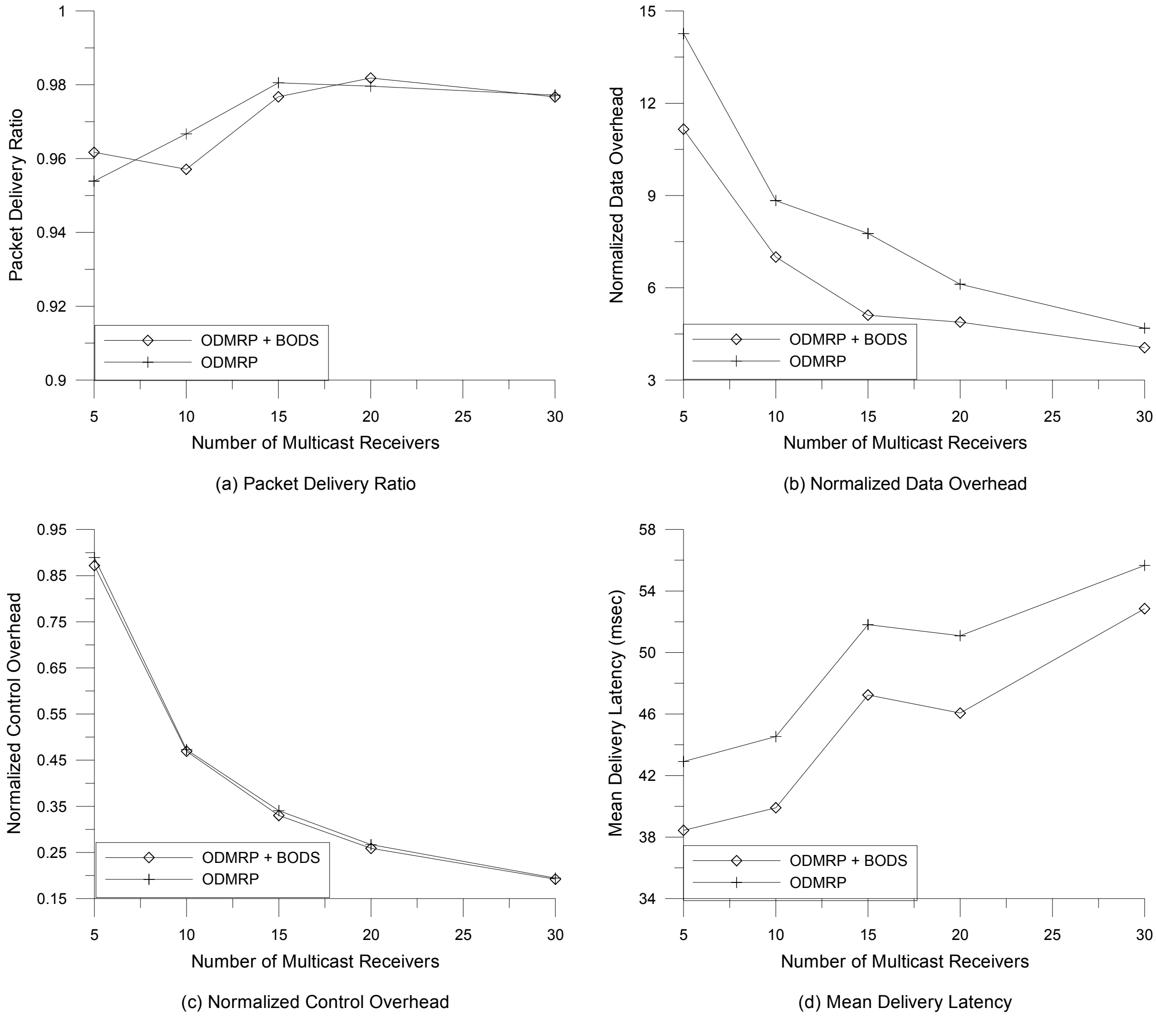


Figure 6.6 Performance versus number of multicast receivers in two-source scenario under RW model.

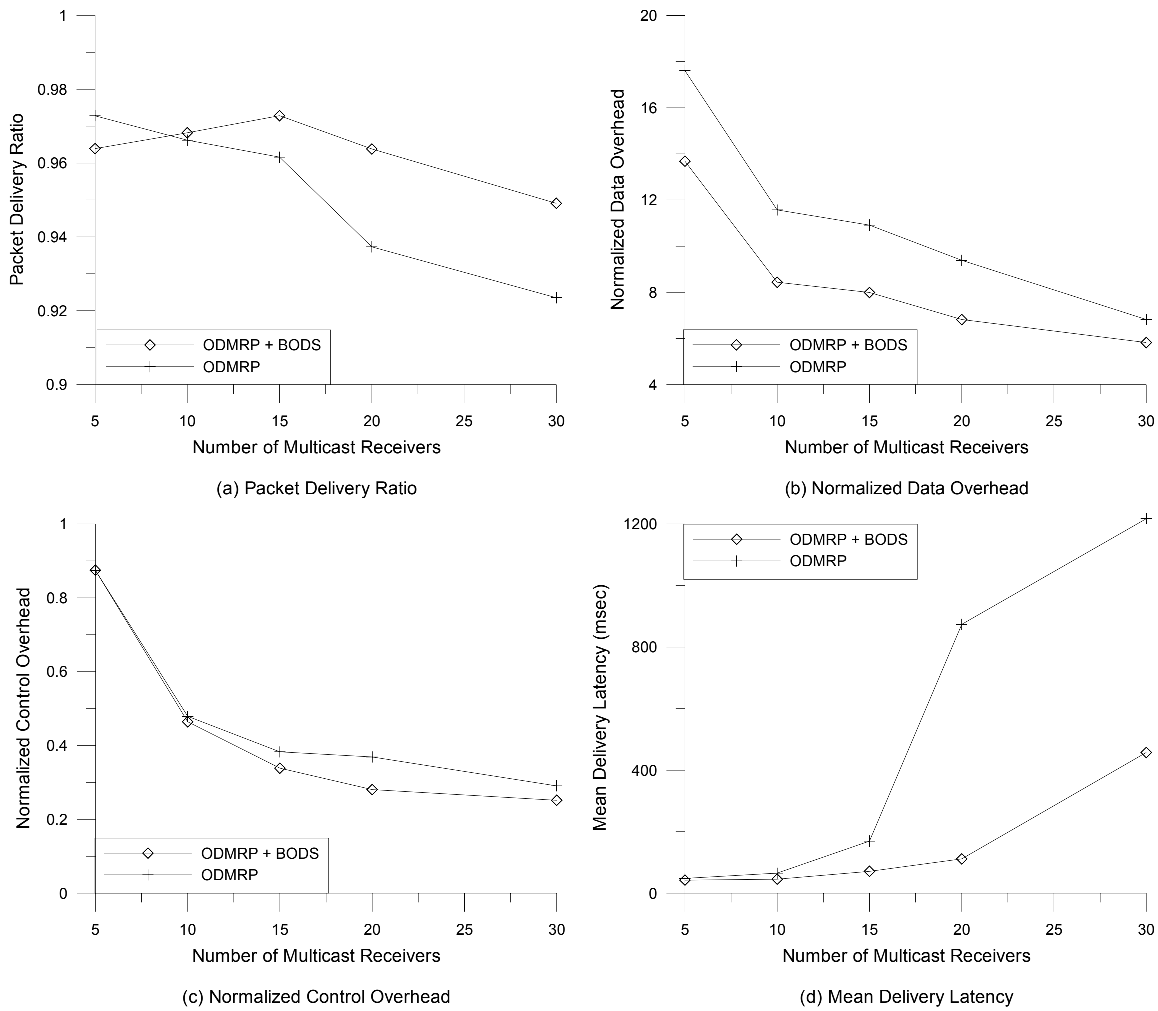


Figure 6.7 Performance versus number of multicast receivers in three-source scenario under RW model.

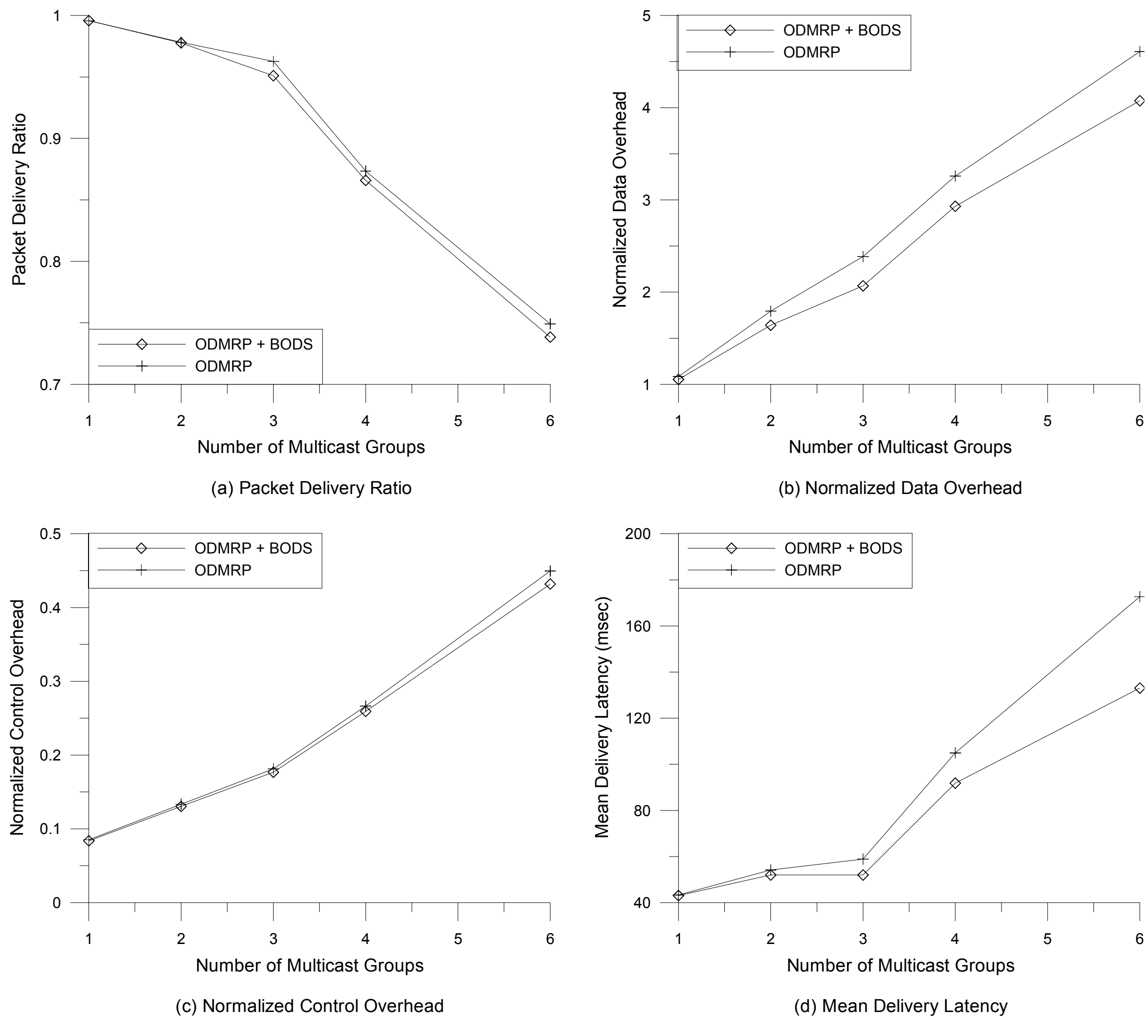


Figure 6.8 Performance versus number of active multicast sessions (1 source per group).

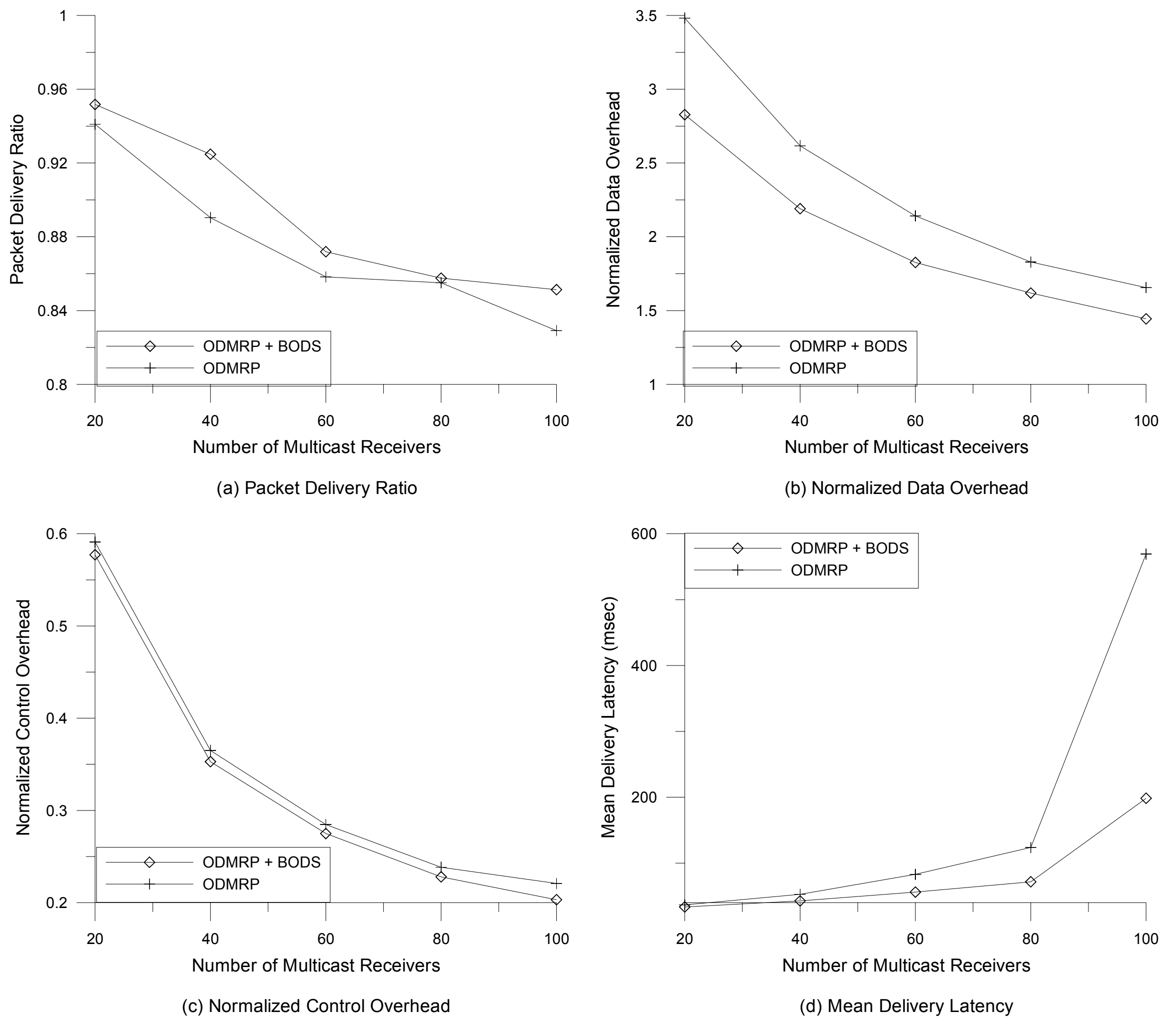


Figure 6.9 **Performance versus number of multicast receivers in four-source scenario under RPGM model.**

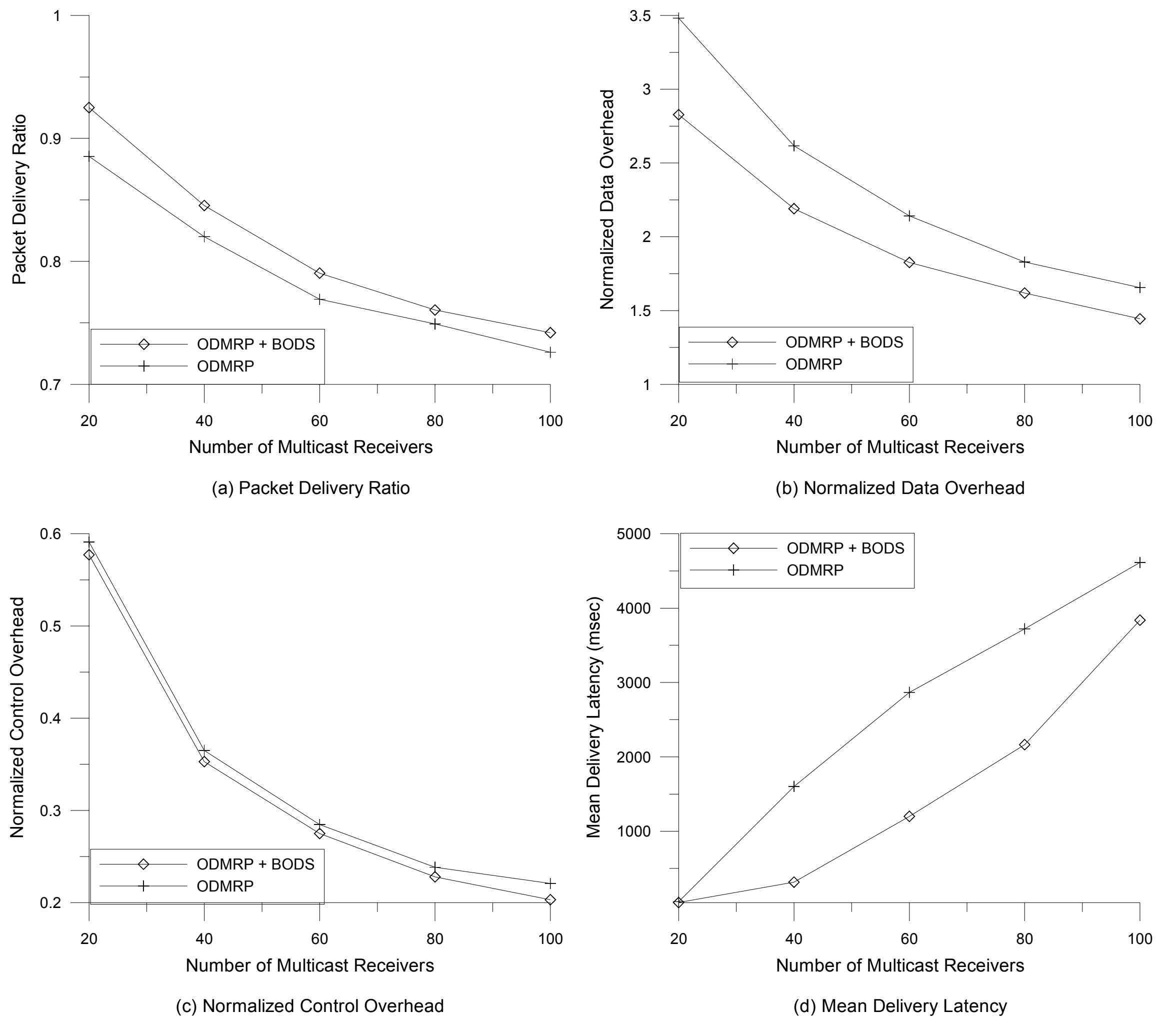


Figure 6.10 Performance versus number of multicast receivers in five-source scenario under RPGM model.

6.5 Summary

In this chapter, the BODS algorithm is proposed to construct per-source mesh-based multicast delivery structure which is more optimal in terms of bandwidth consumption without sacrificing delay performance. The BODS algorithm sets up a more bandwidth-optimal multicast delivery structure based on the Nearest-Participant heuristics. The effectiveness of this algorithm was verified by integrating BODS into ODMRP protocol and validated using Qualnet simulator. The simulation results show that the proposed scheme could achieve similar or better PDR than ODMRP with a reduction of around 15 to 30% of data overhead. The BODS algorithm also significantly improved the delay performance of the network especially under high traffic load. This is particularly important for bandwidth-avid and delay-sensitive applications such as multimedia streaming in a bandwidth-limited mobile ad hoc network.

CHAPTER 7

PERFORMANCE ANALYSIS OF GRAPE

7.1 Introduction

In order to evaluate the performance of GRAPE, we implemented and simulated GRAPE multicast routing protocol in QualNet 3.8 [35], a commercial packet-level network simulator developed by Scalable Network Technologies Inc. This simulator provides a detailed and accurate modeling of the physical, MAC and network operation. We compared the performance of GRAPE with BODS to the performance of ODMRP in a variety of mobility and communication scenarios. ODMRP was chosen as a baseline protocol since it has been shown to outperform other multicast protocols in its class [101]. MobDHop was chosen as the underlying clustering algorithm that forms and maintains stable cluster structure for GRAPE. Since this research emphasized in protocol scalability, the scalability of GRAPE in terms of network density, traffic load, mobility, multicast group size, active multicast sessions and active multicast sources was evaluated via a series of carefully designed, repeatable network scenarios.

7.2 Performance Metrics

A similar set of performance metrics used in the evaluation of the well-established multicast routing protocols for MANETs as suggested in [101][115][122] were used in this research to evaluate and compare the performance of the network-layer multicast solution:

i. *Packet Delivery Ratio* (PDR): The number of data packet successfully delivered to multicast destinations over the number of data packets to be delivered to multicast destinations. The PDR value of "1" means all packets are successfully delivered to all multicast receivers.

ii. *Normalized Data Overhead*: The total number of data packets transmitted by both source node and intermediate nodes over the total number of data packets successfully delivered to multicast destinations. A larger value indicates that the protocol incurs higher data overhead and thus less efficient.

iii. *Total Normalized Overhead*: The total number of all data and routing control packets transmitted by all nodes, divided by the total number of all data packets successfully delivered to multicast destinations. A larger value indicates that the protocol incurs higher overhead and thus less efficient.

iv. *Mean Delivery Latency*: The mean difference between the time at which a data packet is generated and the time at which it is received by the multicast destinations. The mean latency is computed independently for each receiver and then the values are averaged across all multicast receivers.

7.3 Simulation Setup and Protocol Parameters

Table 7.1 and Table 7.2 summarize GRAPE and ODMRP parameters which had been used in all simulations respectively. For the simulations of GRAPE, different redundancy factors (ReFs) i.e. zero redundancy (ReF=0), half redundancy (ReF=0.5) and full redundancy (ReF=1.0), were used in order to evaluate their impact on the performance of GRAPE. For the simulations of ODMRP, we used the default values for ODMRP, which conform to the ODMRP Internet Draft version 4. MobDHop parameters were similar to those presented in Section 4.3.1.

As mentioned, the performance of GRAPE was evaluated using QualNet 3.8. The IEEE 802.11 DCF was used as the MAC protocol while the free space propagation model was used at the radio layer. The communication range was 376m which is the default value for IEEE 802.11 DCF with channel capacity of 2Mbps in QualNet simulator. In all simulation runs, nodes move according to Reference Point Group Model (RPGM) with mobility group of 10 members and maximum group deviation of 400 metres in a 2500 metres x 2500 metres area for 600 seconds of simulated time. The average number of neighbors for each node falls within the range of 13 and 28. Different simulation parameters such as node number, average node speed, and packet generation rate were varied in the simulations in order to evaluate GRAPE effectiveness and scalability. Table 7.3 lists six different simulation configurations to represent different kinds of network and traffic conditions. For each network configuration, ten different scenarios were randomly generated by varying the seed number. Each data point presented in the performance graphs was the average of these ten results.

In all simulations, CBR traffic flows were injected into the network from multicast source nodes for a continuous 300 seconds. The size of data payload was 512 bytes. Multicast sources and receivers were randomly selected among all network nodes. In all simulations, source nodes were also members of multicast group. To better evaluate the effectiveness of GRAPE multicast routing function, membership control features were turned off in all simulations. All group members join the multicast group at the beginning of the simulation and remain as members till the end of the simulation.

Table 7.1 GRAPE, BODS and MobDHop Parameters

GRAPE, BODS and MobDHop Parameter	Value
Source Refresh Interval	20 sec
MG_Flag Timeout	60 sec
Redundancy Factor (ReF)	0, 0.5, 1.0
BODS β	1.0
Delay for highest priority MREQ	4 msec
Delay for intermediate priority MREQ	7 msec
Delay for lowest priority MREQ	10 msec

Table 7.2 ODMRP parameters

ODMRP Parameter	Value
ODMRP Refresh Interval	20 sec
ODMRP FG_FLAG Timeout	60 sec
ODMRP Maximum Retransmission of Join Reply	3
ODMRP ACK for Join Reply Timeout	0.075 sec
ODMRP Aggregation of Join Reply Interval	0.025 sec

Table 7.3 Simulation setup and parameters

	Parameter of Interest in Simulations					
	Network Density	**Average Node Speed**	**Packet Generation Rate**	**Multicast Receivers**	**Multicast Source**	**Multicast Groups**
Node number	{400, 500, 600, 700, 800}	400	400	400	400	400
Number of multicast receivers	20	20	30	{10, 20, 30, 40, 50, 60, 70, 80}	20	20
Number of multicast groups	3	3	1	1	1	{1, 2, 3, 4, 5}
Number of source per group	1	1	1	1	{1, 2, 3, 4, 5}	1
Packets per second	2	2	{2, 5, 8, 10, 16, 20}	2	2	5
Average node speed (m/s)	0	{0, 5, 10,15, 20, 25}	0	{0, 2, 15}	{0, 2, 15}	{0, 2, 15}
Pause time (s)	0	0	0	{0, 100, 0}	{0, 100, 0}	{0, 100, 0}
Purpose: to demonstrate that the protocol scales well w.r.t. parameters of interest	Network density	Mobility rate of network	Traffic load input to network	The number of simultaneous multicast receivers	The number of simultaneous multicast sources	The number of active multicast groups in network

7.4 Simulation Results and Discussions

In this series of simulations, our emphasis was to evaluate how the protocol works and scales with respect to different network parameters of interest such as network density, traffic load, mobility and multicast support. To evaluate the scalability of GRAPE protocol with respect to network density, mobility and traffic load, three sets of simulations have been conducted by varying the number of node, average node speed and CBR traffic generation rate respectively. In these simulations, GRAPE with ReF=0, ReF=0.5, and ReF=1.0 (denoted as GRAPE-0, GRAPE-0.5 and GRAPE-1.0 respectively) were simulated and their performances were compared against ODMRP.

In order to assess the effectiveness of GRAPE with respect to multicast-related parameters, the number of multicast receivers, the number of multicast sources and the number of active multicast groups (sessions) were varied in the subsequent simulations. Furthermore, we conducted these simulations under two different network mobility conditions, i.e. static network and highly mobile network. Most of the existing multicast routing protocols proposed for MANETs over-emphasized the performance in mobile scenarios. The growing interest in wireless mesh networks [142] inspires us to look into the scalability of ad hoc multicast routing protocol in both static and mobile scenarios. In these simulations, we investigated the performance of GRAPE with redundancy factor of 1.0 and compared its performance against the performance of ODMRP.

7.4.1 Network Density

Figure 7.1 shows the performance of both GRAPE and ODMRP when the network density was increased. In this simulation, the number of nodes in the network was varied from 400 to 800 in an area of 2500 metres x 2500 metres. CBR traffic was injected into the network via three multicast sources for three different multicast groups at the rate of 2 packets per second. In each multicast group, 20 multicast receivers were selected randomly to join the multicast group at the beginning of the simulation. To reduce other side effects, no mobility was introduced into the network.

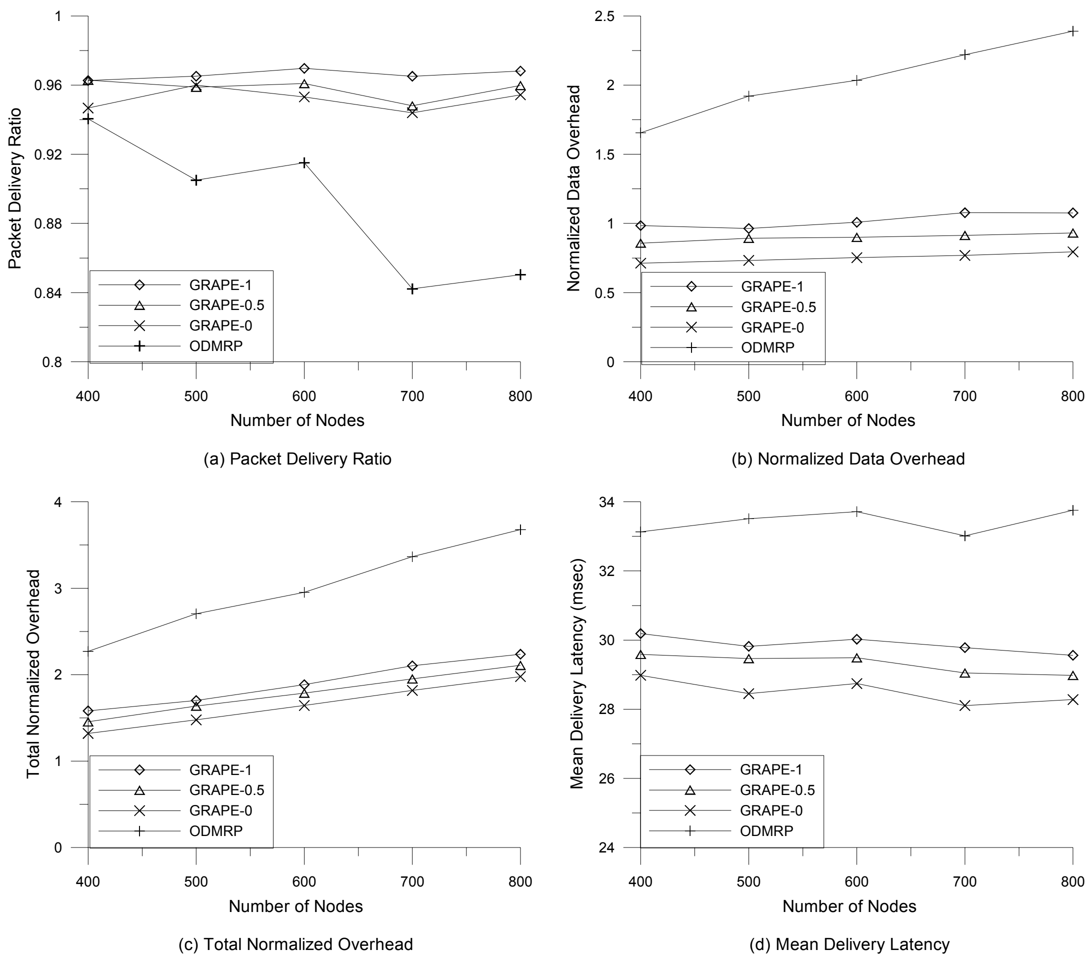


Figure 7.1 Performance versus network density (Group size is 20, 3 groups, 1 source per group).

As shown in Figure 7.1(a), GRAPE-1.0 delivered most multicast data packets (around 96%) to its receivers. Though lower redundancy factor was used, both GRAPE-0.5 and GRAPE-0 achieved high PDR, i.e. around 95%. This is mainly because the network was static. Therefore, the redundancy factor does not play a vital role in this scenario. However, GRAPE-1.0 may implicitly get over the packet loss due to MAC collision problem by sending two copies of similar data packets over the network via two different paths. Due to the lack of mobility in these scenarios, packet losses were incurred due to IEEE 802.11-DCF collisions where three-way handshaking was not implemented for multicast frames and the lost multicast frames were not retransmitted. The PDR of ODMRP dropped below 85% when the network density was increased dramatically. This shows that ODMRP was less scalable in terms of network density. In the contrary, GRAPE scaled better in terms of network density due to its two-tier forwarding mechanism which reduces the number of nodes joining the upper-tier multicast mesh. This reduces the number of forwarding nodes and thus reduces the number of unnecessary duplicated data packets. This can be shown in Figure 7.1(b) where GRAPE incurred 40% to 60% less data overhead than ODMRP. The difference in data overhead incurred was further widened when the network density was increased. Figure 7.1(c) shows the total normalized overhead incurred by the protocols. Although GRAPE assumes MobDHop which requires a periodical Hello message to maintain the cluster structure, the total overhead incurred by all GRAPE variants was still lower than the total overhead incurred by ODMRP. GRAPE outperformed ODMRP in terms of delay performance in spite of the fact that BODS may introduce a small extra delay during the multicast path setup phase. The mean delivery latency of the multicast packets was 10% lower in GRAPE than in ODMRP.

Most of the prior work in multicast routing protocol design were evaluated via simulations in small networks (50 -100 nodes in 1000 metres x 1000 metres of shorter transmission range=250 metres) [95][97][100][101][102][110]. It is unclear how these protocols scale with the network density.

7.4.2 Mobility

Figure 7.2 shows the performance of both GRAPE and ODMRP as a function of mobility. In this simulation, the node average speed was varied from 0 m/s (0 km/h) to 25 m/s (90 km/h). CBR traffic was introduced into the network via three multicast sources for three different multicast groups at the rate of two packets per second. For each multicast group, 20 multicast receivers were selected randomly and they joined the multicast group at the beginning of the simulation. As shown in Figure 7.2(a), PDR degraded with the increased mobility level. This is unsurprising since higher mobility causes higher number of link breaks and inevitable packet loss. This decreasing trend was also observed in [101][115][121]. GRAPE-1.0 achieved the highest PDR in all cases. It delivered 15% more data packets than ODMRP in highly mobile scenario (25 m/s). This can be attributed to the deterministic redundancy it uses in multicast packet forwarding. Since each data packet is delivered to the destination via two different paths, there is higher chance that the packet will arrive at the destination successfully. GRAPE-0.5 and GRAPE-0 also outperformed ODMRP. Although GRAPE-0 only uses single path in multicast delivery, it managed to send about 70% of data packets under highly mobile scenario, which is about 10% higher than the corresponding PDR of ODMRP. It is also shown in Figure 7.2(b) that the data overhead of ODMRP was around two times the data overhead of GRAPE variants. GRAPE-0 incurred the least amount of data overhead since it only used single path to forward data packets. Figure 7.2(d) shows that the mean delivery latency of GRAPE variants was shorter than that of ODMRP by 10%.

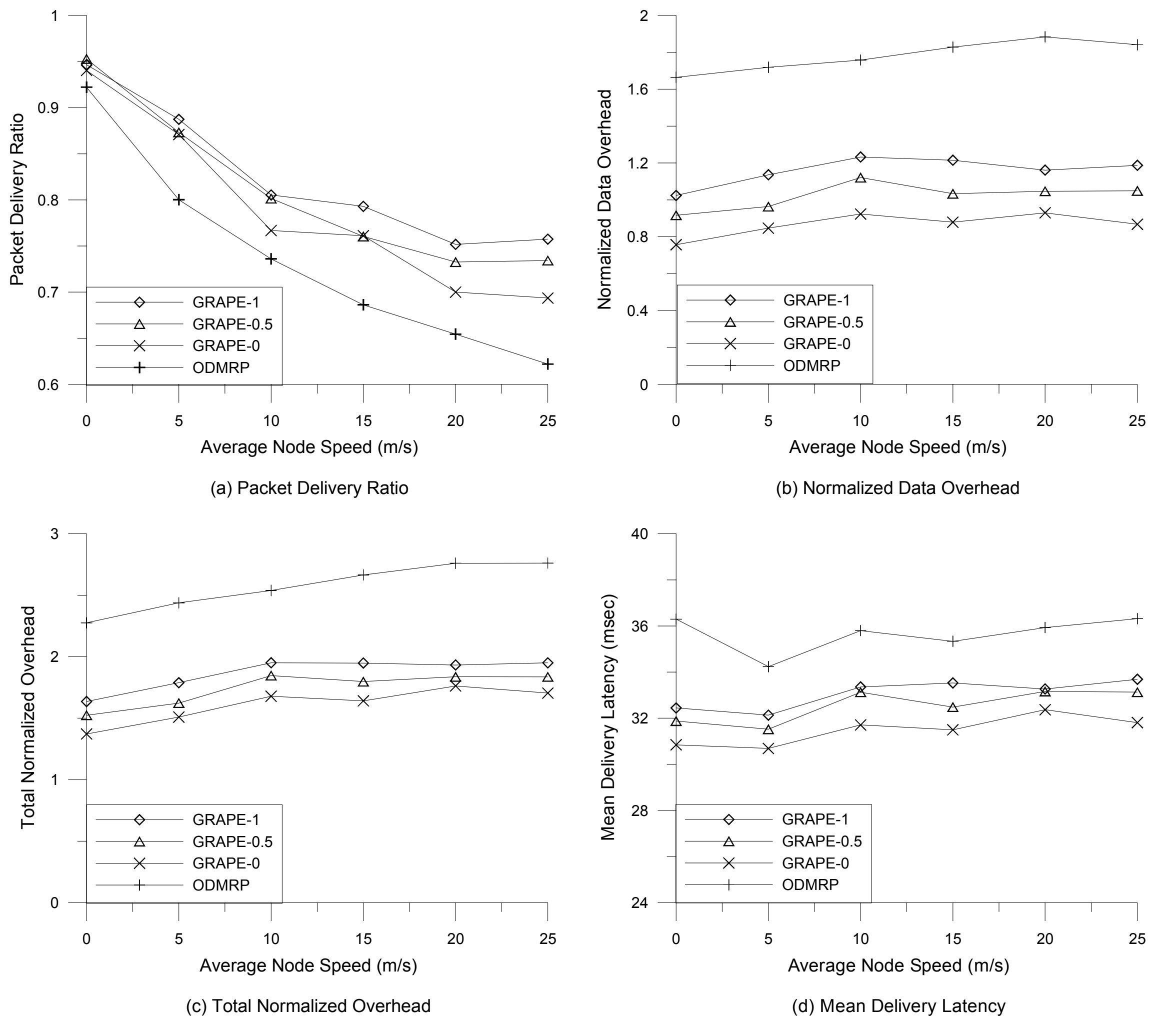


Figure 7.2 Performance versus mobility (Group size is 20, 3 groups, 1 source per group).

7.4.3 Traffic Load

In this set of simulations, our emphasis was to evaluate how the protocol works and scales with respect to the traffic load. A single multicast session was simulated. One source and 20 multicast receivers were chosen randomly to join the multicast group communication. The CBR traffic generation rate was varied from 2 to 20 packets per second. To avoid performance variation caused by mobility, nodes remained static for the entire communication session.

Figure 7.3(a) shows that the PDR of GRAPE variants and ODMRP degraded with the increased traffic load. However, ODMRP suffered a sharper decrease due to a larger number of redundant data packets that consequently clogged the network.

Although GRAPE-1.0 offers path redundancy, the performance degradation when traffic load grows quickly was not as serious as in ODMRP. The data overhead incurred by GRAPE variants was much lower than the data overhead incurred by ODMRP. This is mainly attributed to the multicast delivery mesh constructed by BODS is more optimal in terms of forwarding efficiency and the reduced number of nodes joining the upper-tier mesh construction. The normalized data overhead and control overhead were lower at higher traffic load. This indicates that the increment in the amount of both types of overhead is slower than the growth in traffic rate. An additional benefit of reducing the data overhead is the reduction of the mean delivery latency. Since ODMRP constructs a shortest-path mesh, it is expected to offer a better delay performance. However, the heavy contention in MAC layer causes the lengthening of delay in ODMRP. This claim can be validated by the increasing trend observed in the mean delivery latency of ODMRP as shown in Figure 7.3(d). Meanwhile, all GRAPE variants gave a better delay performance due to the reason that the collisions and MAC contentions are indirectly minimized as a result of lesser data transmission due to a lighter and more efficient forwarding delivery infrastructure constructed.

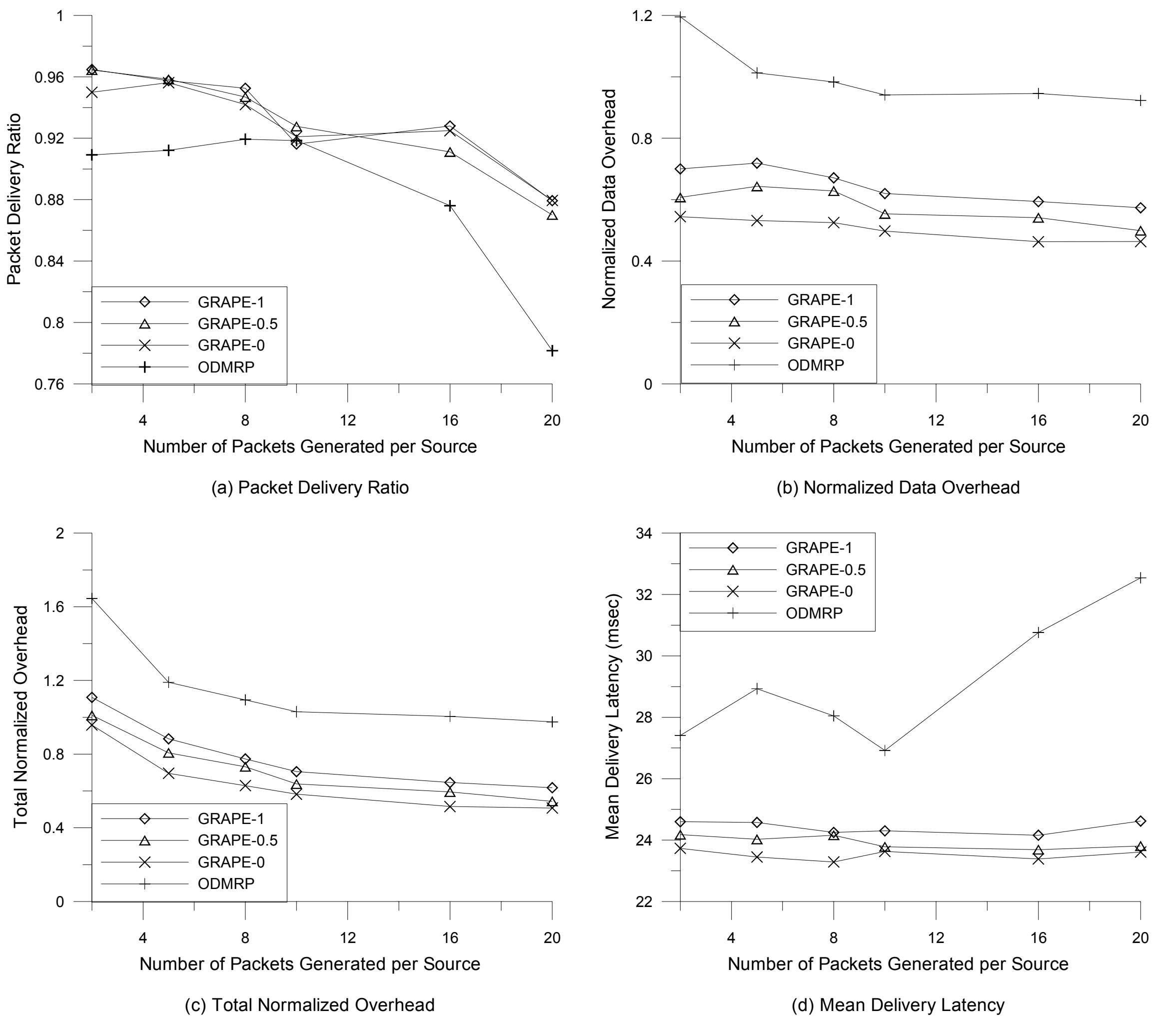


Figure 7.3 Performance versus traffic load (Group size is 30, 1 group, 1 source per group).

7.4.4 Multicast Scalability

This section was aimed to evaluate the protocol effectiveness and efficiency in supporting different multicast scenarios. Hence, three multicast-related parameters were varied in these sets of simulations, i.e. the number of multicast receivers (group size), the number of multicast sources and the number of active multicast sessions. It was shown in previous simulations that GRAPE-1.0 outperformed GRAPE-0.5 and GRAPE-0 with respect to the robustness in multicast packet delivery. Therefore, only GRAPE-1.0 was simulated here and its performance was compared against ODMRP which is a robust mesh-based multicast protocol as documented in literature

[101][121]. We also tested different multicast requirements in both static and highly mobile networks in order to assess their adaptability against different network mobility. In highly mobile scenarios, nodes moved according to RPGM model at an average speed of 15 m/s continuously.

7.4.4.1 Number of Multicast Receivers

Figure 7.4 and Figure 7.5 show the performance metrics as functions of group size in static, semi-static and highly mobile scenario respectively. In these simulations, one multicast session was simulated with single source node, which generated 2 multicast packets per second. The number of multicast receivers was varied from 10 to 80.

In static scenario, the PDR of both GRAPE-1.0 and ODMRP as shown in Figure 7.4(a) decreased slightly with the increase of group size. As the network was static, the packet loss was mainly due to MAC layer collisions. MAC layer collisions were more severe when the number of group member increased. This is due to the fact that, as the larger fraction of network nodes were included in the multicast data delivery structure, the higher chance that MAC layer contention and collision would take place. GRAPE-1.0 outperformed ODMRP in terms of protocol robustness and efficiency in the static scenario. A higher PDR was achieved due to the more efficient multicast delivery structure formed by BODS and the reduced sets of participating nodes (only clusterhead of clusters with multicast group members). The delay performance of GRAPE-1.0 was also better than that of ODMRP.

In highly mobile scenario, GRAPE-1.0 also outperformed ODMRP in terms of PDR as shown in Figure 7.5(a). It is observed that both GRAPE-1.0 and ODMRP delivered a larger fraction of packets as group size increases. This is because the forwarding mesh formed by both GRAPE-1.0 and ODMRP becomes more reliable as more network nodes were included in the forwarding mesh due to a larger number of multicast receivers. A similar trend was also observed in [121] and [115] when the authors simulated the performance of ODMRP. GRAPE-1.0 is capable of achieving better PDR in highly mobile scenario by using much lower data overhead as shown in Figure 7.5(b) and (c). The novel design of GRAPE that uses two-tier forwarding mechanism and deterministic path redundancy contributes to the superiority of GRAPE over ODMRP.

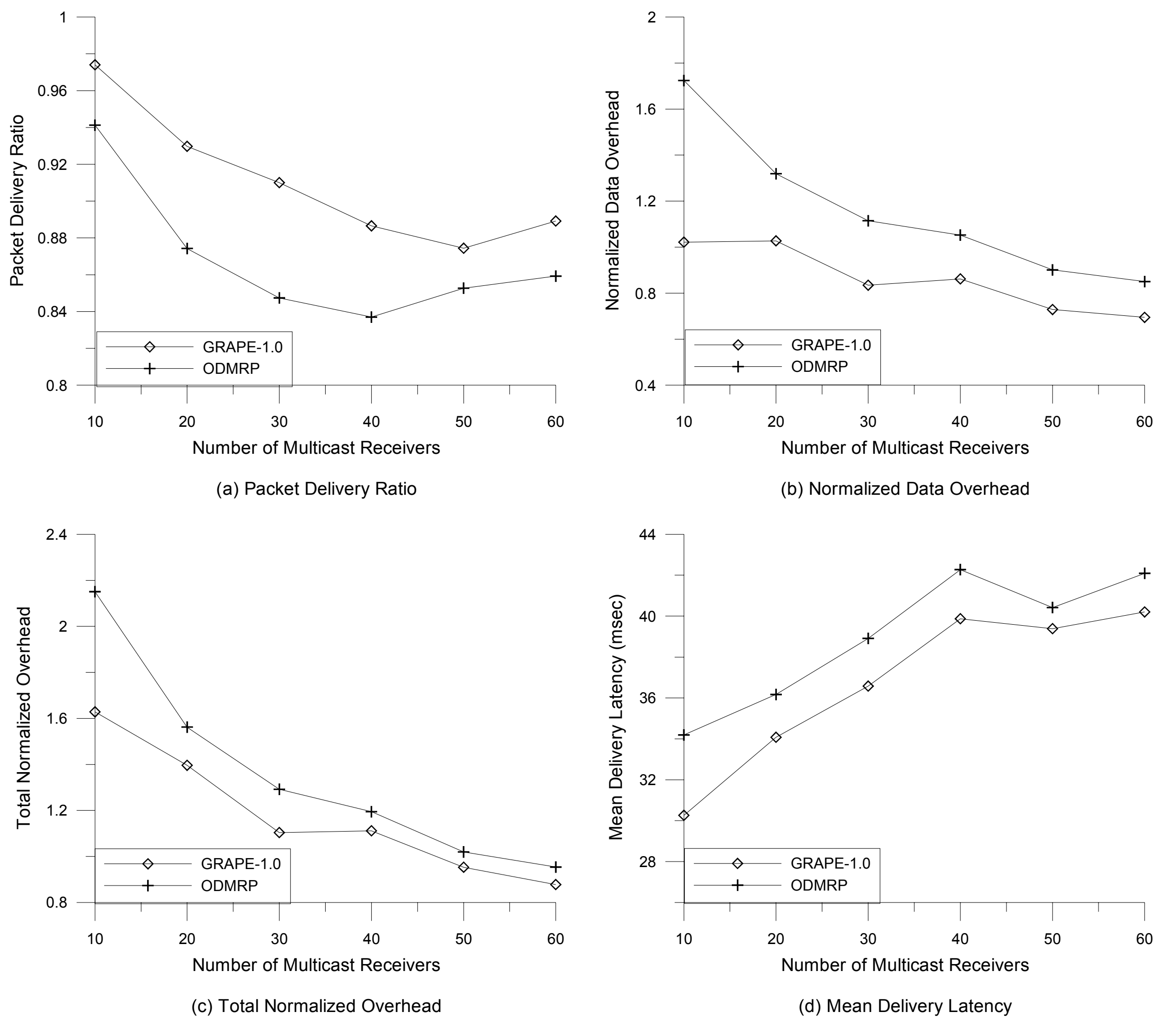


Figure 7.4 Performance versus multicast group size in static scenario (1 group, 1 source per group).

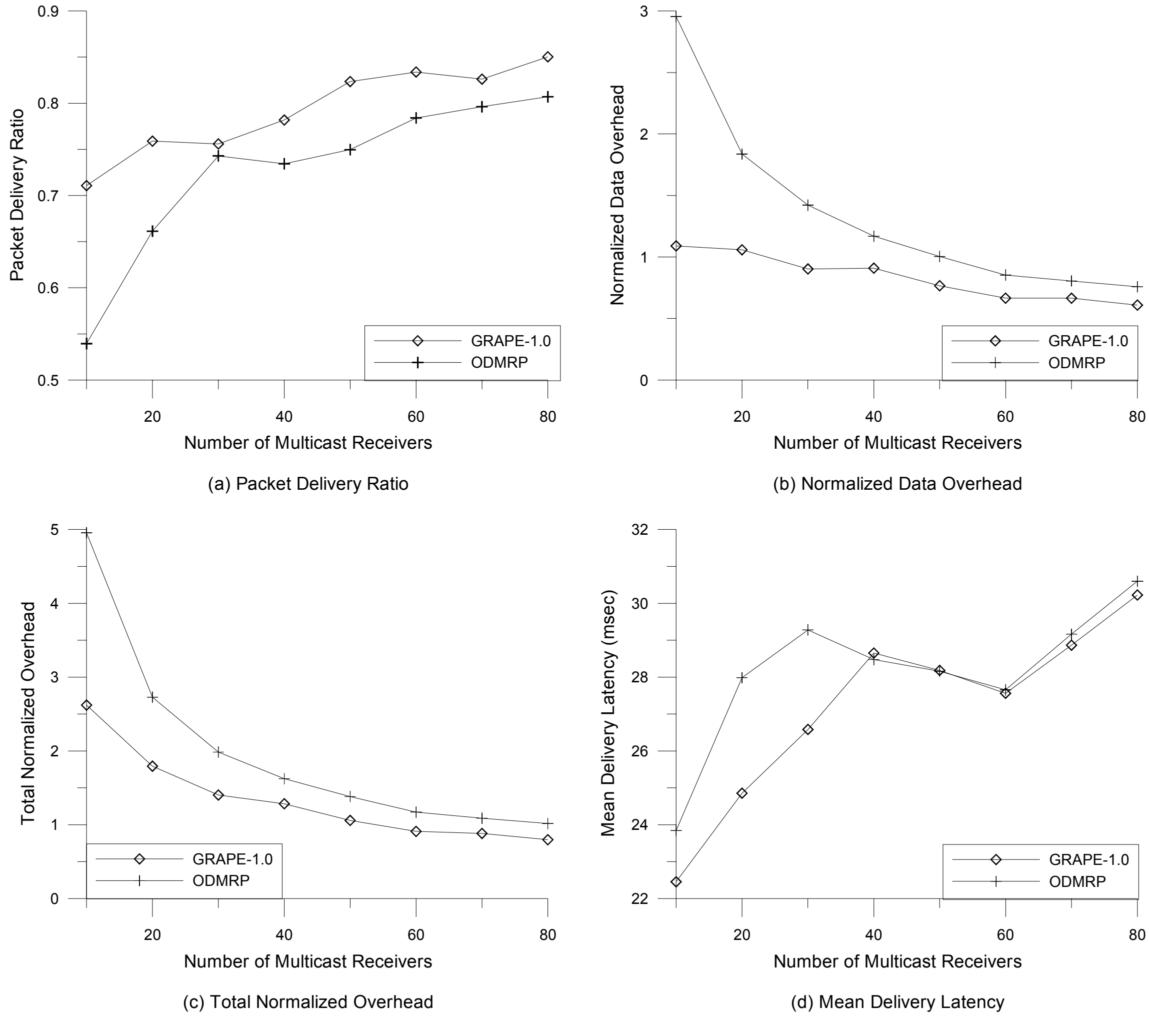


Figure 7.5 Performance versus multicast group size in highly mobile scenario (1 group, 1 source per group).

Table 7.4 summarizes some previously reported results where the authors simulated their protocol and compared the performance of their protocol against ODMRP. Since HDDM [115] is a hierarchical multicast protocol, it was simulated in relatively larger mobile networks. HDDM was able to outperform ODMRP in terms of PDR when the group size was small. However, the performance of HDDM degraded when the group size increased. It should be noted that HDDM incurred much longer mean delivery latency due to the use of much longer path.

Table 7.4 A summary of previously reported results in literature by varying group size

Protocol	Simulator	Scenario	Observations
MAODV [121]	NS-2	50 nodes 1000m x 1000m RW 20.83 m/s	PDR degraded with the increase of group size and consistently lower than PDR of ODMRP. The difference became larger when the group size increased. Delay performance was not presented.
DDM [95]	NS-2	50 nodes 1000m x 1000m RW 0 - 2 m/s	PDR degraded significantly with the increase of group size and consistently lower than PDR of ODMRP. It was more efficient in terms of data overhead when the group size is small. Delay performance was not presented.
HDDM [115]	GloMoSim	400 nodes 2500m x 2500m RW 1 - 20 m/s	PDR degraded slightly with the increase of group size. HDDM outperformed ODMRP when the group size was small. It was more efficient in terms of both data and control overhead. However the delay was two times the delay of ODMRP.
DCMP [111]	GloMoSim	50 nodes 1000m x 1000m RW 0 - 20 m/s	Comparable PDR in large and small multicast groups. More efficient in terms of control and data overhead. Delay performance was not presented.

7.4.4.2 Number of Multicast Sources

Figure 7.6 and Figure 7.7 show the performance metrics as functions of the number of multicast sources in both static and highly mobile scenario respectively. In these simulations, one multicast session was simulated with 20 multicast receivers randomly chosen to join the multicast communication. The number of multicast sources was varied from 1 to 6. Each source generated two multicast packets per second.

Figure 7.6(a) depicts the PDR of both GRAPE-1.0 and ODMRP in a static scenario. Both protocols delivered more than 90% of data packets in one-source, two-source, three-source and four-source scenarios. However, the PDR dropped below 90% when the number of sources was increased to five and six. The decreasing trend was mainly due to the fast-growing traffic level introduced by the increasing number of sources. The network became highly congested and the number of packet loss due to collisions increased. A similar observation was made by Ji and Corson [95]. Figure 7.6(b) shows that GRAPE-1.0 achieved better PDR than ODMRP in static scenario while incurring 30% less data overhead. Although ODMRP was capable of delivering comparable number of multicast packets to its receivers, the mean delivery latency increased significantly with the number of multicast sources in the network. In six-source scenario, the mean delivery latency of ODMRP was about 200 milliseconds, which may jeopardize the performance of multimedia and voice applications. Meanwhile, GRAPE-1.0 delivered packets within 100 milliseconds in all cases.

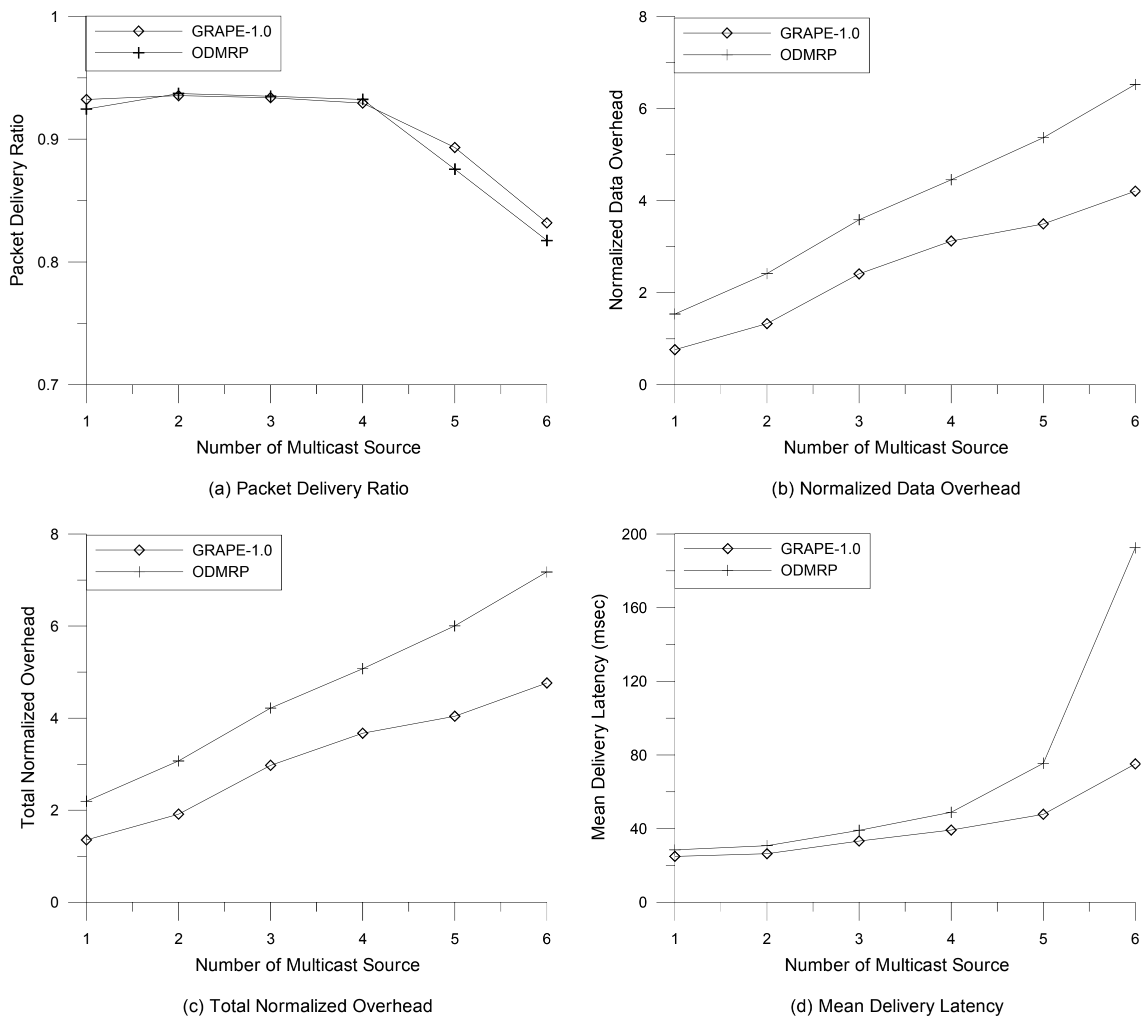


Figure 7.6 Performance versus number of multicast sources in static scenario (Group size is 20, 1 group).

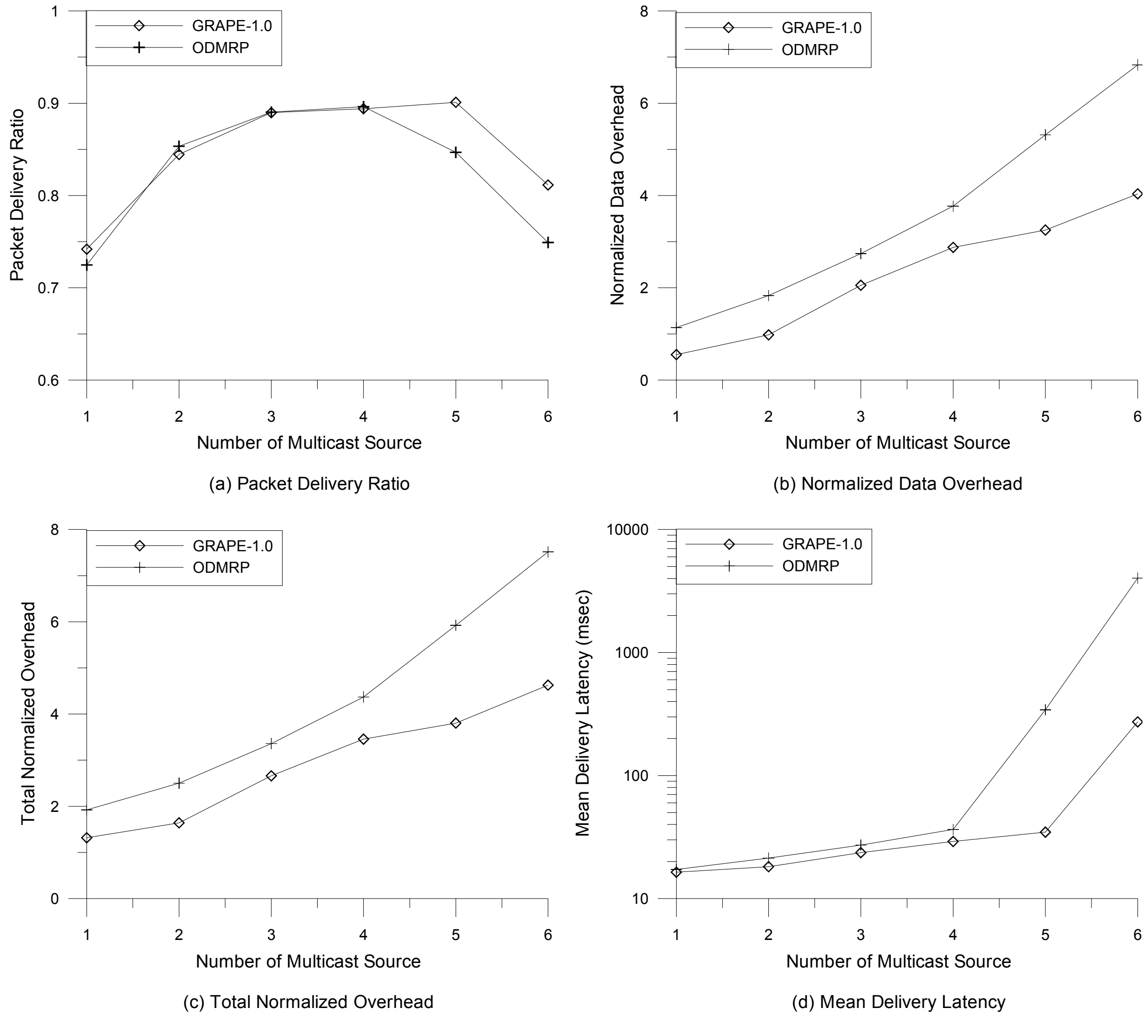


Figure 7.7 Performance versus number of multicast sources in highly mobile scenario (Group size is 20, 1 group).

Figure 7.7(a) shows the PDR of both GRAPE-1.0 and ODMRP in a highly mobile scenario. GRAPE-1.0, again, delivered more multicast packets to the receivers than ODMRP. In one-source scenario, both GRAPE-1.0 and ODMRP could only deliver about 75% of multicast packets. This is mainly due to the high mobility rate in the network that caused frequent link breaks and thus higher packet loss. The PDR of both GRAPE-1.0 and ODMRP increased with the number of sources since larger number of paths were found and incorporated into the mesh formed by both protocols. This, in turns, increased the robustness of both protocols. However, the PDR of both protocols decreased when the number of sources increased to five and six respectively. The reason is the network became too busy and congested due to a much higher offered load injected to the network by increased number of multicast sources. A similar observation has been done by the authors of DDM protocol [95]. Nevertheless, GRAPE-1.0 was able to maintain 80% of PDR in six-source scenario while the PDR of ODMRP dropped below 80%. The mean delivery latency of ODMRP (i.e. 4000 milliseconds) was about 20 times higher than that of GRAPE-1.0 (i.e. 200 milliseconds) in six-source scenario.

Table 7.5 presents a summary of three previously reported results where the authors evaluated their protocol by varying the number of multicast sources and compared the performance of their protocol against ODMRP. It is observed that MAODV [121] and DDM [95] failed to outperform ODMRP in terms of PDR. Although DCMP [111] could achieve comparable PDR as ODMRP, the performance of DCMP in terms of mean delivery latency remains unclear.

Table 7.5 A summary of previously reported results in literature by varying the number of multicast sources per group

Protocol	Simulator	Scenario	Observations
MAODV [121]	NS-2	50 nodes 1000m x 1000m RW 20.83 m/s	PDR increased slightly with the number of sources but it was still lower than that of ODMRP in all cases tested. Delay performance was not presented.
DDM [95]	NS-2	50 nodes 1000m x 1000m RW 0 - 2 m/s	PDR degraded faster than ODMRP with the increase of the number of sources. Delay performance was not presented.
DCMP [111]	GloMoSim	50 nodes 1000m x 1000m RW 0 - 20 m/s	Comparable PDR in large and small multicast groups. More efficient in terms of control and data overhead. Delay performance was not presented.

7.4.4.3 Number of Multicast Sessions

Figure 7.8 and Figure 7.9 show the performance metrics as functions of the number of simultaneous multicast sessions in static and highly mobile scenario respectively. In these simulations, each multicast session was simulated with one source and 20 multicast receivers. The number of multicast sessions was varied from 1 to 5. Each multicast source generated five multicast packets per second.

In the static scenario as shown in Figure 7.8, the PDR of GRAPE-1.0 was consistently higher than that of ODMRP for about 5% while incurring 30% less data overhead and 10% shorter mean delivery latency. It is observed that both protocols are less sensitive to the increase in the number of multicast sessions. Traffic load was increased with the number of simultaneous multicast sessions. However, it might be introduced evenly into the entire network without stressing any particular wireless link unlike the case as observed in Section 7.4.4.2 where the number of multicast sources was increased. Although the traffic load was increased, network congestion was not observed in these simulations unlike the case in Section 7.4.4.2 where the network congestion was observed when the number of sources was increased beyond 4. This can be further validated by the relatively more stable delay performance that was demonstrated by both protocols in Figure 7.8(d).

A decreasing trend in PDR was observed in highly mobile scenario as shown in Figure 7.9. Both GRAPE-1.0 and ODMRP could only send 70% of data packets in network with one multicast session. This ratio decreased to 63% (GRAPE-1.0) and 59% (ODMRP) respectively when the number of multicast session was increased to five. Gui and Mohapatra [115] also observed a decreasing trend in terms of PDR when they simulated the performance of ODMRP in a mobile network. However, in their scenario, traffic load was maintained at the same rate for all scenarios regardless of the number of multicast sessions running. Therefore, it may support our observation that the mobility is the main factor in these simulations that caused the decrease in PDR.

Table 7.6 summarizes previously reported results in the literature where the authors tested their protocol against ODMRP by varying the number of active multicast sessions. HDDM outperformed ODMRP in terms of PDR in their simulation. However, it incurred 0.8 times higher mean delivery latency than that of ODMRP, which may make it unsuitable for some delay-sensitive applications. Yi et al [14] also evaluated their protocol, M-LANMAR by varying the number of multicast sessions. It demonstrated a stable performance in terms of PDR for all tested scenarios and it outperformed ODMRP when the number of multicast sessions was more than five. However, it is unclear how M-LANMAR performs in terms of routing overhead and delay.

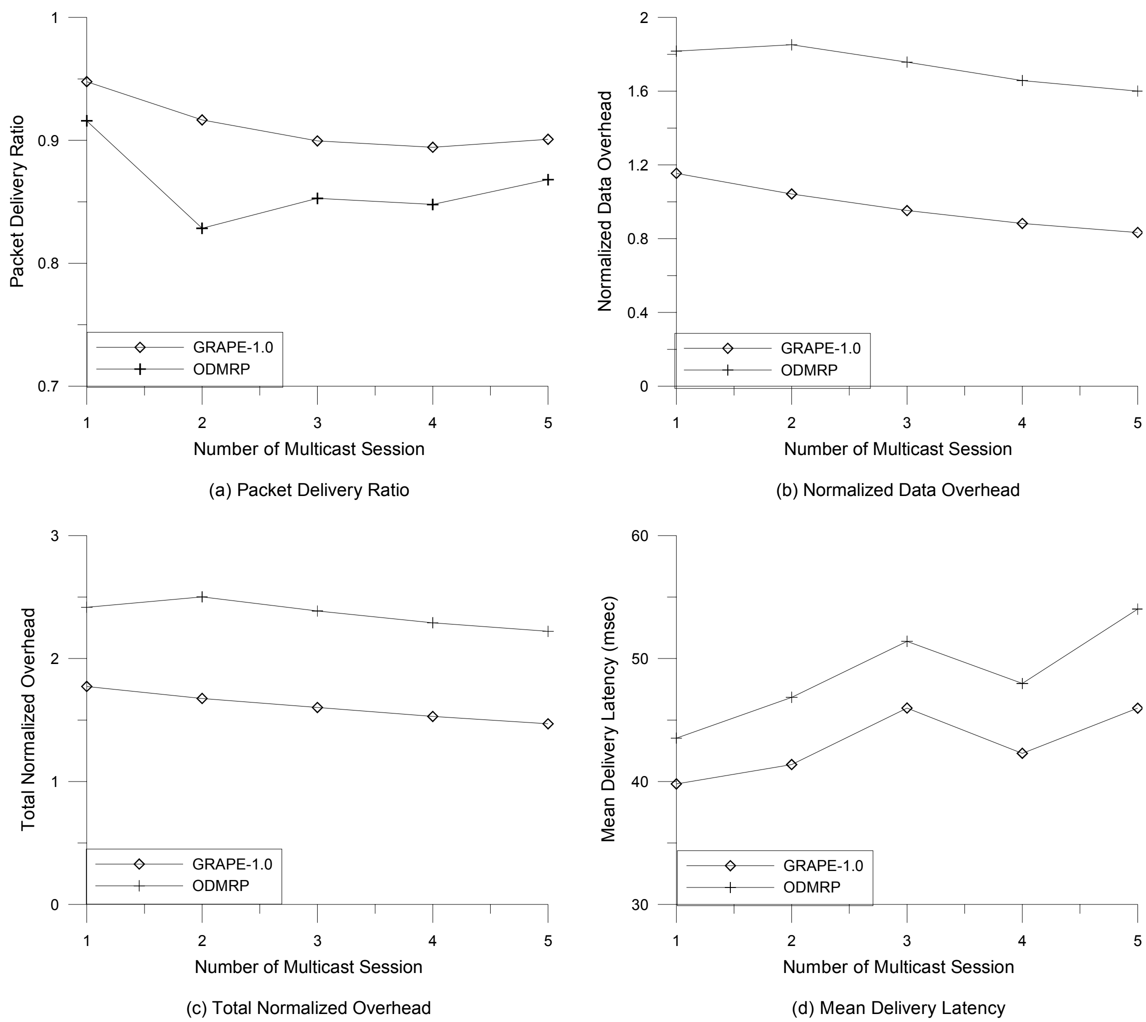


Figure 7.8 Performance versus number of multicast sessions in static scenario (Group size is 20, 1 source per group).

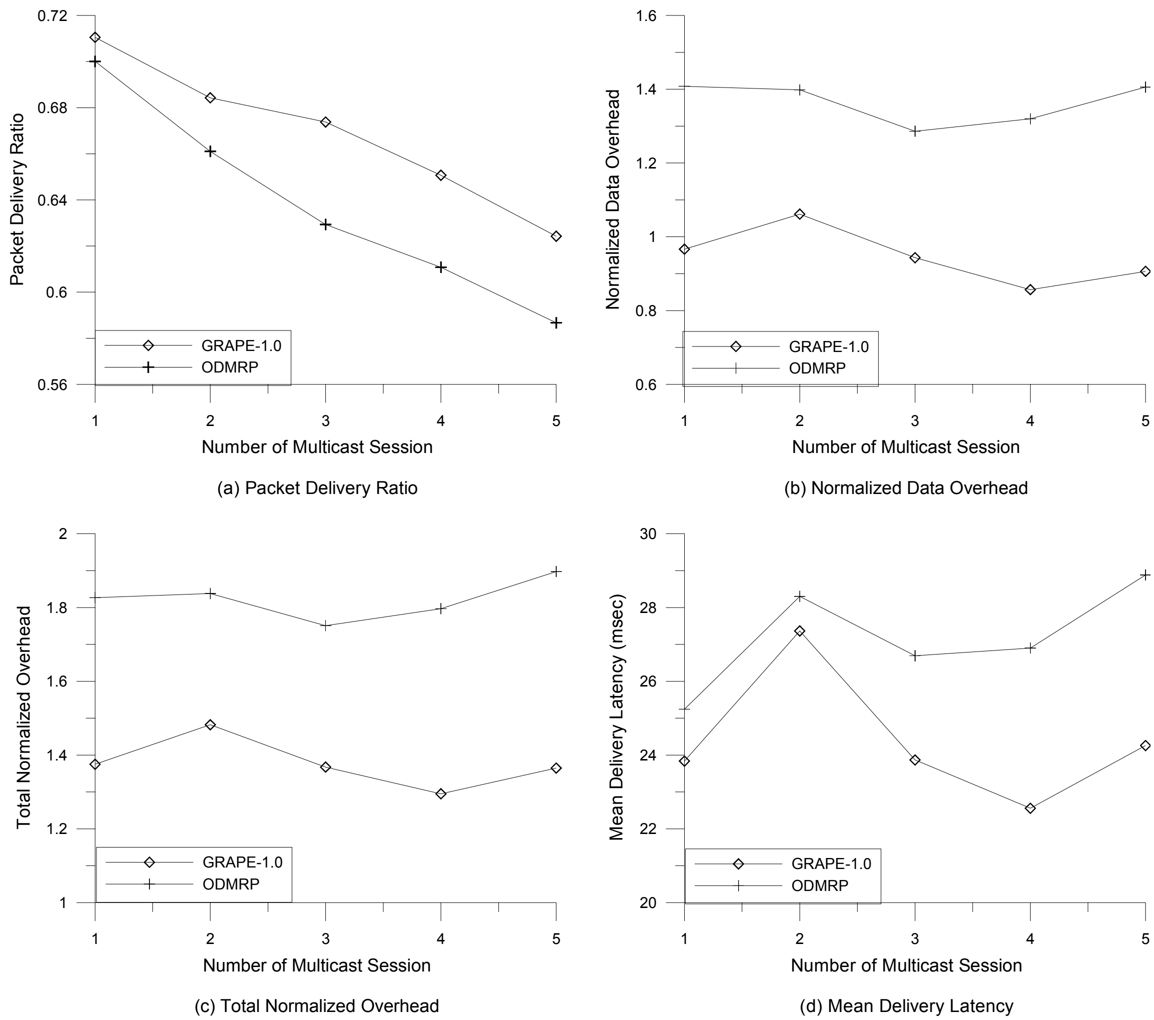


Figure 7.9 Performance versus number of multicast sessions in highly mobile scenario (Group size is 20, 1 source per group).

Table 7.6 A summary of previously reported results in literature by varying the number of simulataneous multicast sessions

Protocol	Simulator	Scenario	Observations
MAODV [121]	NS-2	50 nodes 1000m x 1000m RW 20.83 m/s	PDR was lower than that of ODMRP in two test cases. Delay performance was not presented.
HDDM [115]	GloMoSim	400 nodes 2500m x 2500m RW 1 – 20m/s	HDDM outperformed ODMRP in terms of PDR in all test cases. It is more efficient in terms of overhead. However the mean delay is about 0.8 times of the mean delay of ODMRP.
M-LANMAR [14]	QualNet	1000 nodes 6000m x 6000m RPGM 2 m/s	Stable PDR performance for all different test cases. Delay and overhead performance were not presented.

7.5 Summary

The performance of GRAPE was assessed through an extensive series of simulations in QualNet 3.8. Its performance was compared quantitavely against the performance of ODMRP based on four performance metrics, namely PDR, normalized data overhead, total normalized overhead and mean delivery latency under similar network configurations. It is important for a multicast routing protocol to deliver all multicast packets to all multicast receivers with a short latency. Meanwhile, the amount of data and control overhead incurred by the protocol might limit their scalability in terms of network density, traffic load and other multicast-related parameters. A protocol that incurs a large amount of overhead may waste the scarce bandwidth of MANETs and hence does not scale well.

GRAPE delivered more packets to destinations than ODMRP in most, if not all, scenarios at the expense of much lower data overhead. An additional benefit of GRAPE is the better delay performance in all tested scenarios, which is particularly important in delay-sensitive applications. GRAPE scales gracefully with respect to network density, mobility and traffic load. GRAPE-1.0 is most robust among the three GRAPE variants simulated here. Therefore, it is suitable for application that requires reliable packet delivery. Some previous reported results are summarized and presented in this chapter to provide insights into how other proposed multicast routing protocols perform with respect to different simulation parameters.

CHAPTER 8

CONCLUSION AND FUTURE WORK

8.1 Summary of Findings

The main objectives of this research were to: (i) to design a clustering algorithm for MANETs that can adapt to mobility pattern and form stable cluster structure to support network control functions and (ii) to design a multicast routing protocol for MANETs that can fully utilize the pre-existing stable, two-tier control structure and achieve desirable multicast efficiency, robustness against mobility and protocol scalability.

Figure 8.1 shows the multicast architectural design of IP multicasting. Meanwhile Figure 8.2 depicts a flat multicast architectural design which is usually assumed in the design of multicast routing protocol in MANETs. In this research, the architectural design of multicast solution is different from both IP multicasting and flat MANET multicasting. As shown in Figure 8.3, our design was based on a two-tier logical hierarchy. Therefore, this research consisted of two main parts. First, a mobility-adaptive multihop clustering algorithm, MobDHop, has been proposed to provide a long-lived and efficient cluster structure in support of scalable two-tier multicast routing purposes. Second, a cluster-based, GRoup-AdaPtivE multicast routing solution, GRAPE, has been proposed to provide scalable multicast routing solution that delivers multicast data packets robustly and efficiently across mobile ad hoc networks of different configurations. To further enhance the multicast capability of MANETs, a new, general multicast path setup algorithm, namely Bandwidth-Optimized and Delay-Sensitive (BODS) algorithm has also been proposed in this reseach to construct a more optimal multicast delivery structure in terms of bandwidth utilization.

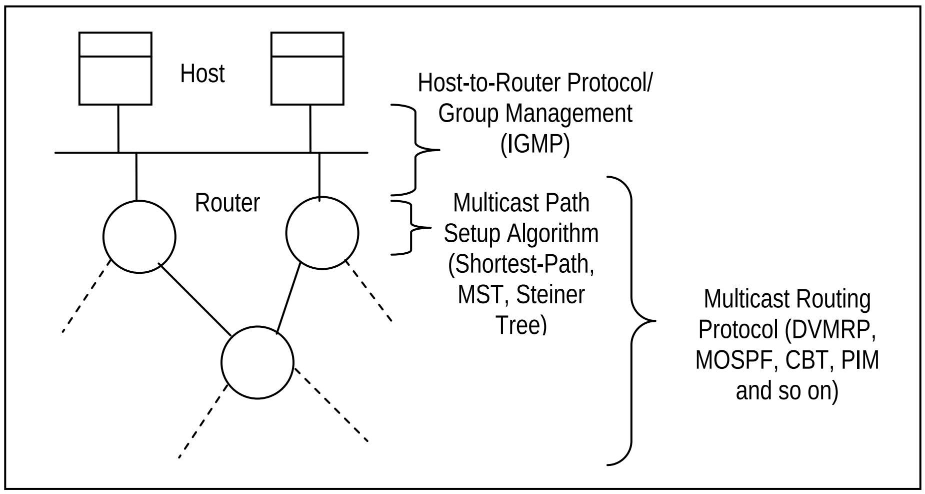


Figure 8.1 Architectural design of IP multicasting.

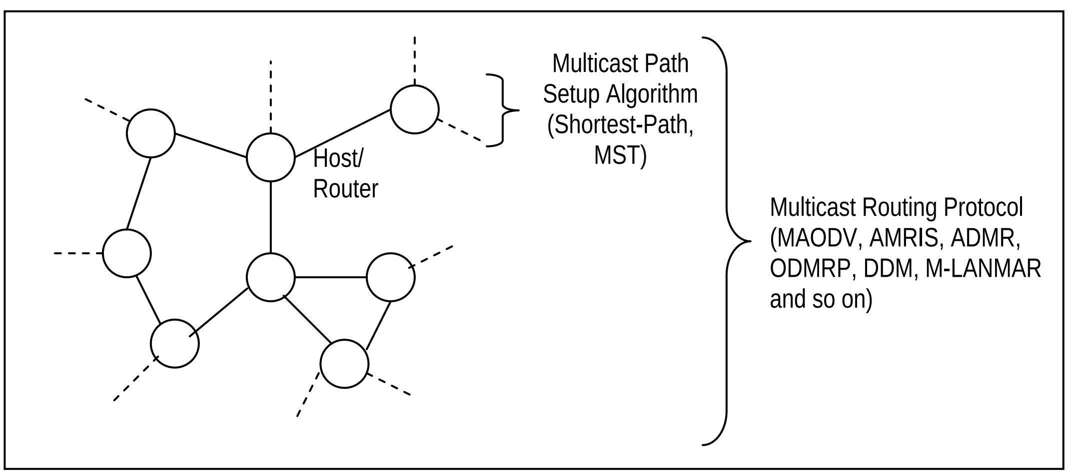


Figure 8.2 Architectural design of flat MANET multicasting.

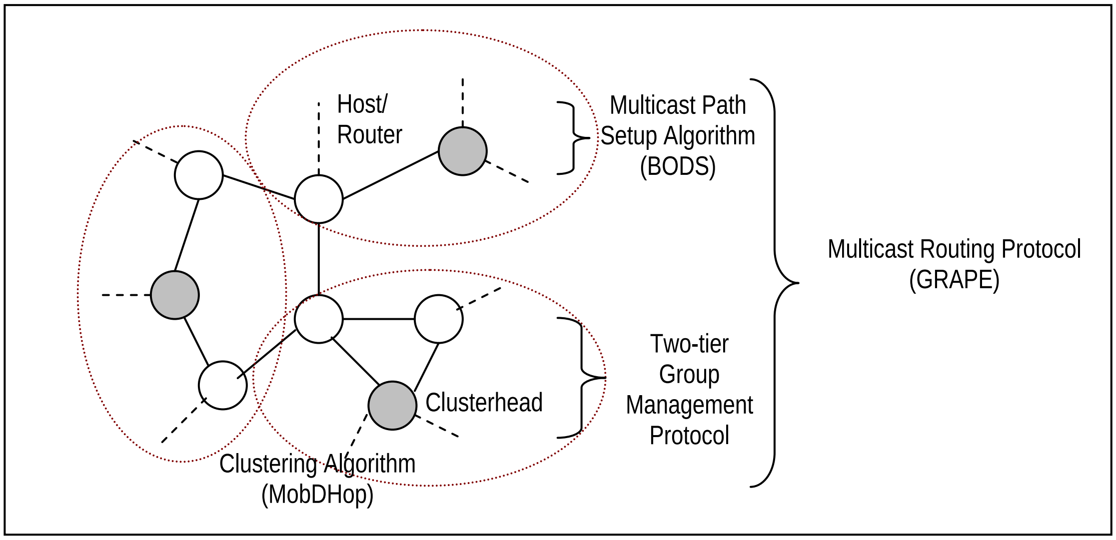


Figure 8.3 Architectural design of two-tier multicasting in this research.

8.1.1 Mobility-based D-Hop (MobDHop) Clustering Algorithm

Mobility-based D-Hop (MobDHop) clustering algorithm has been proposed in this research to form stable and efficient cluster structure for different kinds of network scenario by taking the mobility pattern into consideration during cluster formation and cluster maintenance. MobDHop was modeled and simulated using NS-2 simulator along with two well-known clustering algorithms which are Lowest-ID (L-ID) clustering and Maximum Connectivity Clustering (MCC) in different network scenarios by varying different network parameters such as network area, node number, node average speed, and node mobility pattern.

Simulation results show that the performance of MobDHop, L-ID and MCC were comparable when the random mobility model was used in relatively small network (100 nodes). This is because the clustering is almost impossible in a random network without coordinated movement. However, the performance of MobDHop was much better than L-ID and MCC when the group mobility model was used. The cluster formed by MobDHop conformed to the grouping pattern in the network scenario if the grouping pattern exists in the scenario. This could be attributed to the use of a simple mobility-based metric, local variability value, as clusterhead election criteria in MobDHop algorithm. Therefore, MobDHop could capture group mobility pattern in the network if any exist. Meanwhile other algorithms use metrics such as node identifier and node degree that do not take mobility effects into direct consideration during cluster formation.

Besides, the impact of clustering for different group communication patterns in large and sparse networks was also investigated by simulations. Apart from L-ID and MCC, different variations of MobDHop were simulated by varying an important parameter in MobDHop, which is the maximum hop count from the clusterhead. Results show that MobDHop again outperformed other clustering algorithms in terms of cluster stability and cluster efficiency for different kinds of group communication scenarios in larger and sparser MANETs. This may be mainly attributed to the ability of MobDHop to form multihop clustering that allows group members that are more than one-hop away from their clusterhead to join the appropriate cluster. Appropriate clustering may further reduce the clusterhead changes and cluster re-affiliation events in the network considerably. These properties usually lead to a more long-lived cluster structure. Hence, in large and sparse networks, MobDHop is without any doubt superior to what can be found in the L-ID and MCC to form a stable cluster structure in order to support efficient and scalable multicast routing function. It was also observed in the simulation results that unnecessary multihop clustering in small group may deteriorate the stability of the cluster structure.

Unlike other multihop clustering algorithms, the maximum hop count parameter in MobDHop, namely d can be set to a very large value in MobDHop. Even if d is larger than the network diameter, MobDHop will not form unreasonably large clusters as in other multihop clustering algorithms. d is an important parameter in other multihop clustering algorithms that must be predefined before the execution of the algorithm in order to limit the multihop clusters from growing too large. This is not the case in MobDHop since MobDHop utilizes cluster stability information to form stable multihop clusters while d is primarily a limiting factor that can be set to meet network management requirements. When the stability criterion is not met during the merging phase, cluster will not grow and remain in its most stable state. Hence, MobDHop can adaptively form variable-hop clusters which are more stable based on the use of local variability metric to identify and capture group mobility patterns in MANETs. A stable cluster structure is essentially important and useful for spatial reuse, Quality of Service support, network management and security provision.

The analysis of message and time complexity of MobDHop gave an insight into how MobDHop reacts to topology changes. This analysis shows that the number of packet transmissions per node per time step required for MobDHop to operate correctly in MANETs is O(1). We provide the upper bound of time complexities for both cluster formation and cluster maintenance in MobDHop. It is shown that multihop clustering is feasible in networks with high mobility without incurring prohibitive signalling overhead.

It is also shown in this thesis that MobDHop clustering algorithm can be used to support unicast routing functionality. A new variant of AODV protocol, namely MobDHop-AODV is proposed to utilize the stable, two-tier cluster structure formed by MobDHop algorithm in order to reduce network-wide flooding of control messages.

8.1.2 GRoup-AdaPtivE (GRAPE) Multicast Routing Protocol

A stable cluster structure lends itself to the design of a scalable multicast routing solution in MANETs. In this research, a complete network-layer multicast solution has been proposed, consisting of three main components: (1) multicast path setup algorithm, (2) multicast group management mechanism and (3) multicast routing protocol.

The new multicast path setup algorithm, BODS algorithm, constructs a source-based, bandwidth-optimal multicast delivery structure based on Nearest-Participant Heuristic without sacrificing delay performance. Being a multicast path setup algorithm, BODS is a general algorithm that can be integrated into any existing source- and mesh-based multicast routing protocols. The effectiveness of BODS was first evaluated by integrating BODS into an multicast routing protocol, namely the On-Demand Multicast Routing Protocol (ODMRP). ODMRP is a source- and mesh-based multicast routing protocol which was reported in the literature to be very robust against mobility. The performance results, obtained from simulations in the QualNet simulator, revealed that BODS-integrated ODMRP achieved a similar or better PDR as compared to the original ODMRP with a reduction of around 30% data overhead. The BODS algorithm also improved the delay performance of the network especially under high traffic loads.

The GRoup-AdaPtivE (GRAPE) multicast routing protocol has been proposed in this reseach to introduce a new two-tier multicast group management scheme and a novel two-tier multicast routing protocol. GRAPE utilizes the cluster structure formed by the MobDHop clustering algorithm to provide efficient and effective multicast routing function over a relatively larger MANETs. In GRAPE, a new multicast group management scheme that distributes the management load to all clusterheads in the network has been proposed. Apart from this, GRAPE also introduces a two-tier multicast routing hierarcy that supports multicast routing with desirable properties, such as high multicast efficiency, high robustness against mobility, and more scalable. The upper-tier multicast communication structure connects multicast source to clusterheads that are interested to join the multicast communication on behalf of their members. The packet dissemination for upper-tier structure is done in a more efficient, source- and mesh-based multicast delivery structure which is constructed by using the BODS algorithm. The lower-tier multicast communication infrastructure connects each clusterhead and its members. A clusterhead dynamically select a suitable forwarding scheme, i.e. either cluster broadcasting or stateless multicasting to forward packets to its members based on the traffic characteristic within their clusters. It may switch from one scheme to another if the traffic within cluster changes. For example, when more than 40% of cluster members join the same multicast group, a clusterhead will broadcast every data packet from this multicast group to its cluster members. Otherwise, a DDM-like tree-based multicasting will be adopted. The robustness of multicast routing is further enhanced in GRAPE by introducing the multi-path property which is widely used in unicast routing. A redundancy factor is introduced in order to provide deterministic path redundancy in

GRAPE. In short, GRAPE offers a complete network-layer multicast routing solution that has multi-path property that can be used in various kinds of MANET configuration and applications.

The effectiveness and benefits of GRAPE were validated by simulations for different network conditions and multicast requirements. The performance of GRAPE was compared against the performance of ODMRP under similar simulation settings. Results show that GRAPE delivered more packets to destinations than ODMRP in most, if not all, scenarios as well as incurring much lower data overhead. The better delay performance of GRAPE over ODMRP makes GRAPE a better alternative in delay-sensitive applications. GRAPE scales gracefully with respect to network density, mobility and traffic load as shown in simulation results. GRAPE-1.0, where GRAPE delivers each data packet via two distinct paths, is the most robust scheme among three GRAPE variants (GRAPE-1.0, GRAPE-0.5 and GRAPE-0) simulated (cf: Section 7.4.1 to 7.4.3). Therefore, it is more suitable for applications that require reliable packet delivery.

In conclusion, GRAPE promises a better alternative to the network industry in their process to extend multicast capability to the existing MANET protocol stack. GRAPE offers a more efficient and robust multicast mechanism which is suitable for large mobile ad hoc networks and large multicast applications. The two-tier multicast packet delivery structure formed by GRAPE is simple and could be implemented easily without much modification to the existing protocol stack. Besides, GRAPE could work on other pre-existing logical or physical cluster structure as well. However, the performance of GRAPE is greatly correlated to the stability and efficiency of the underlying cluster structure provided. Therefore, it is recommended that, GRAPE should be generalized to other cluster structures, which is stable and efficient in order to guarantee optimum protocol performance.

8.2 Future Work

In this section, we discuss some aspects of this research that may need further study and can thus become potential future work

8.2.1 Mobility-based D-Hop (MobDHop) Clustering Algorithm

It was observed in the MobDHop performance studies that unnecessary multihop clustering in small group may deteriorate the stability of the cluster structure due to a relatively short merge interval. Therefore, a adopting a longer merge interval may help to improve the stability of cluster structure formed by MobDHop when smaller groups dominate in the network. However, it was also observed that using a longer merge interval will cause most nodes to stay un-clustered for a longer period of time and therefore the network control functions that use the cluster structure may be affected since valid a cluster structure is harder to achieve in this case. Hence, this is a trade-off in algorithm design that should be further investigated. It is also possible to investigate other evaluation methods to analyze the performance of different clustering algorithms via a theoretical perspective. Previous work on the analysis of control packet overhead incurred by clustering algorithm is mainly focused on the derivation of control overhead in the big-O notation with respect to network size. This may not be adequate as various other network parameters will affect the volume of control overhead generated, e.g. node mobility, node transmission range, and network density. Analysis of clustering control overhead that takes into account node mobility, network size and network density [143] will be very helpful in refining the future design of clustering algorithm. Another possible extension is to apply competitive analysis, which is widely used in online algorithms, to compare the performance of different distributed online clustering algorithms in MANETs.

8.2.2 Bandwidth-Optimized and Delay-Sensitive (BODS) Algorithm

The BODS algorithm can also be integrated into other multicast routing protocols (besides ODMRP) in MANETs. It is also beneficial if the performance of BODS can be analysed via a theoreotical perspective. Competitive analysis which is commonly used in the analysis of centralized online algoirithm might be extended to evaluate the theoretical performance of BODS. However, this is challenging since BODS works in a fully distributed manner and the network environment varies over time.

8.2.3 GRoup-AdaPtivE (GRAPE) Multicast Routing Protocol

The use of clusterheads to manage group membership and as the forwarders of multicast packets may result in clusterheads becoming bottlenecks or hot spots in GRAPE multicast routing due to their extensive in the multicast packet forwarding infrastructure. This situation could happen when the traffic load in the network is high. High traffic load may cause congestion at immediate links which are connected to clusterheads since clusterheads are in-charge of forwarding all packets for their cluster members. Therefore, some load balancing mechanisms should be designed to divert data packets from the clusterhead in order to prevent it from becoming the hotspots or bottlenecks. A possible solution is to limit the number of nodes that a clusterhead can handle by imposing a cluster size parameter as limiting factor.

8.2.4 Future Work

Currently, GRAPE is assumed to work on top of a cluster structure formed by MobDHop. Its performance with other underlying clustering algorithms should also be evaluated. Alternatively, MobDHop can also be applied to other flat MANET routing protocols in order to improve their performance and scalability. The extent to which these goals can be achieved needs to be studied together with the amount of modifications to the protocols that are needed. Moreover, GRAPE may be further enhanced to support QoS and guaranteed multicast delivery by introducing admission control, data buffering and positive or negative acknowledgement mechanisms.

Although the protocol evaluation via simulations is a widely-accepted practice in the field of network research, the protocol evaluation of MobDHop, BODS and GRAPE would be more useful and industry-relevant if they can be tested in real network scenarios. This could be done by setting up a test-bed, consisting of mobile devices implementing both MobDHop and GRAPE in order to verify their effectiveness in different real-life network scenarios. Besides, drafting GRAPE into an Internet-Draft which is regularly discussed by the Internet Task Force Group will be very useful for the future enhancement and improvement of GRAPE by other researchers in this field. An Internet-Draft is also very useful for further adoption of GRAPE as an industry standard by different mobile device manufacturers.

BIBLIOGRAPHY

[1] *Mobile Ad hoc Networks (MANET)*. MANET Charter. Cited 09 August 2006. Available from: <http://www/ietf.org/html.charters/manet-charter.html>.

[2] Chakrabarti, S. and A. Mishra. QoS Issues in Ad Hoc Wireless Networks, IEEE Communications Magazine, pp. 142-148. 2001.

[3] Kahn, E. Advances in Packet Radio Technology, Proc. of the IEEE, *66(11)*, pp. 1468-1496. 1978.

[4] Belding-Royer, E.M. Routing Approaches in Mobile Ad hoc Networks. In Mobile Ad Hoc Networking, ed by S. Basagni, M. Conti, S. Giordano, I. Stojmenovic, New York: Wiley-IEEE Press. 2004.

[5] Beyer, D.A. Accomplishments of the DARPA SURAN Program. In Proc. of IEEE MILCOM '90, September 1990, Monterey, California, pp. 855-862.

[6] Leiner, B.M., R.J. Ruth and A.R. Sastry. Goals and Challenges of the DARPA GloMo Program, IEEE Personal Communications, *3(6)*, pp. 34-43. 1996.

[7] Zieniewicz, M.J., D.C. Johnson, D.C. Wong and J.D. Flatt. The Evolution of Army Wearable Computers, IEEE Pervasive Computing, *1(4)*, pp. 30-40. 2002.

[8] Erwin, S.I. *Land Warriors' Link Up With Stryker Vehicles*. National Defense Magazine, Cited 09 August 2006. Available from: <http://www.nationaldefensemagazine.org/issues/2006/may/LandWarrior.htm>.

[9] *The Thinking Car*. Siemens AG, Cited 09 August 2006. Available from: <http://www.siemens.com/index.jsp?sdc_p=ft4mls4u20o1361958ni1148625pFEcz3&sdc_sid=20902049449&>.

[10] Huang, E., W. Hu, J. Crowcroft and I. Wassell. Towards Commercial Ad hoc Network Applications - A Radio Dispatch System. In Proc. ACM MobiHOC 2005, March 2005, Urbana-Champaign, IL, USA, pp. 355-365.

[11] Papavassiliou, S., S. Xu, P. Orlik, M. Snyder and P. Sass. Scalability in Global Mobile Information Systems (GloMo): Issues, Evaluation Methodology and Experience, Wireless Networks, *8(6)*, pp. 637-648. 2002.

[12] Gerla, M. Scalable Ad hoc Networking Routing: From Battlefield to Vehicle Grids and Pervasive Computing. In Proc. 4th Annual Mediterranean Workshop on Ad Hoc Networks (Med-Hoc-Net 2005), June 2005, France, pp. 355-365.

[13] Lee, S. J., E.M. Belding-Royer and C.E. Perkins. Scalability Study of the Ad hoc On-Demand Distance Vector Routing Protocol, Int. Journal of Network Management, *13(2)*, pp. 97-114. 2003.

[14] Yi, Y., M. Gerla and K. Obraczka. Scalable Team Multicast in Wireless Ad Hoc Networks Exploiting Coordinated Motion, Ad Hoc Networks 2, pp. 171-184. 2004.

[15] Deering, S. Multicast Routing in a Datagram Network. Ph.D Thesis, Stanford University. 1991.
[16] Maufer, T. and C. Semeria. Introduction to IP Multicast Routing, Internet Draft, Internet Engineering Task Force (1997). draft-ietf-mboned-intro-multicast-03.txt.
[17] Cormen, T.H., C.E. Leiserson, R.L. Rivest and C. Stein. Introduction to Algorithms. pp. 561-620, Cambridge: MIT Press, 2nd edition. 2001.
[18] Hwang, F., D. Richards and P. Winter. The Steiner Tree Problem. Amsterdam: North-Holland. 1992.
[19] Paul, P. and S. Raghavan. Survey of Multicast Routing Algorithms and Protocols. In Proc. 15th International Conference on Computer Communication (ICCC 2002), August 2002, Mumbai, pp. 902-926.
[20] Pusateri, T. Distance Vector Multicast Routing Protocol, RFC 1075, Internet Engineering Task Force (1998). draft-ietf-idmr-dvmrp-v3-06.txt.
[21] Moy, J. Multicast Extensions to OSPF (MOSPF), RFC 1584, Proteon Inc. (1994).
[22] Ballardie, A.J., P. Francis and J. Crowcroft. Core Based Trees (CBT). In Proc. ACM SIGCOMM '93, September 1993, San Francisco, CA, pp. 85-95.
[23] Fenner, B., M. Handley, H. Holbrook and I. Kouvelas. Protocol Independent Multicast (PIM) - Dense Mode: Protocol Specification (Revised), Internet Draft, Internet Engineering Task Force (2003). draft-ietf-pim-sm-v2-new-07.txt.
[24] Adams, A., J. Nicholas and W. Siadak. Protocol Independent Multicast (PIM) - Sparse Mode: Protocol Specification (Revised), Internet Draft, Internet Engineering Task Force (2003). draft-ietf-pim-dm-new-v2-03.txt.
[25] Varshney, U. Multicast over Wireless Networks, Communications of the ACM, *45(12)*, pp. 31-37. 2002.
[26] Chiang, C.C. and M. Gerla. On-Demand Multicast in Mobile Wireless Networks. In Proc. 6th International Conference of Network Protocols, October 1998, Austin, TX, pp. 260-270.
[27] Obraczka, K., K. Viswanath and G. Tsudik. Flooding for Reliable Multicast in Multihop Ad Hoc Networks, Wireless Networks 7, pp. 627-634. 2001.
[28] Sinha, P., R. Sivakumar and V. Bharghavan. MCEDAR: Multicast Core-Extraction Distributed Ad hoc Routing. In Proc. IEEE WCNC'99, September 1999, New Orleans, LA, pp. 1313-1317.
[29] Arpacioglu, O., T. Small and Z.J. Haas. Notes on Scalability of Wireless Ad hoc Networks, IEEE ANS WG Internet Draft (2003). draft-irtf-ans-scalability-definition-01.txt.
[30] Kochhal, M., L. Schwiebert, S. Gupta and C. Jiao. An Efficient Core Migration Protocol for QoS in Mobile Ad Hoc Networks. In Proc. 21st IEEE International Performance Computing, and Communications Conference (IPCCC), April 2002, Phoenix, pp. 387-391.
[31] Mieghem, P., G. Hooghiemstra and R. Hofstad. On the Efficiency of Multicast, IEEE/ACM Transactions on Networking, *9(6)*, pp. 719-732. 2001.
[32] Santivanez, C.A., B. McDonald, I. Stavrakakis and R. Ramanathan. On the Scalability of Ad Hoc Routing Protocols. In Proc. IEEE INFOCOM 2002, June 2002, New York, pp. 1688-1697.
[33] Gupta, S.K.S. and B. Wang. Energy-Efficient Multicast Protocols. In Resource Management in Wireless Networking, ed by M. Cardei, I. Cardei and D.Z. Du, Netherlands: Kluwer Academic Publishers. 2005.
[34] Fall, K. and K. Varadhan. *The ns Manual*. The VINT Project (2002). Cited 09 August 2006. Available from: <http://www.isi.edu/nsnam/ns/>.
[35] *QualNet Network Simulation Software*. Scalable Network Technologies. Cited 09 August 2006. Available from: <http://www.scalable-networks.com>.
[36] Er, I.I. and W.K.G. Seah. Mobility-Based D-Hop Clustering Algorithm for Mobile Ad Hoc Networks. In Proc. IEEE Wireless Communications and Networking Conference (WCNC2004), March 2004, Atlanta, Georgia, pp. 2359-2364.

[37] Er, I.I. and W.K.G. Seah. Performance Analysis of Mobility-based d-Hop (MobDHop) Clustering Algorithm for Mobile Ad Hoc Networks. To appear in Computer Networks.
[38] Er, I.I. and W.K.G. Seah. Clustering Overhead and Convergence Time Analysis of the Mobility-based Multihop Clustering Algorithm for Mobile Ad Hoc Networks. To appear in *Journal of Computer and System Science* (JCSS), special issue on Performance Modeling and Evaluation of Computer Systems.
[39] Er, I.I. and W.K.G. Seah. Distributed Steiner-Like Multicast Path Setup for Mesh-based Multicast Routing in Ad Hoc Networks. In Proc. of the IEEE International Workshop on Multimedia Technology and Ubiquitous Computing (MTUC 2006), in conjunction with the IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC 2006), June 2006, Taichung.
[40] Lin, C.R. and M. Gerla. Adaptive Clustering for Mobile Wireless Networks, IEEE Journal on Selected Areas in Communications, *15(7)*, pp. 1265-1275. 1997.
[41] Ramanathan, R. and M. Steenstrup. Hierarchically-Organized, Multihop Mobile Wireless Networks for Quality-of-Service Support, Mobile Networks and Applications, *Vol. 3*, Baltzer Science Publishers, pp. 101-119. 1998.
[42] Chen, W., N. Jain and S. Singh. ANMP: Ad Hoc Network Management Protocol, IEEE Journal on Selected Area in Communications, *17(8)*, pp. 1506-1531. 1999.
[43] Steenstrup, M. Cluster-Based Networks. In Ad Hoc Networking, ed by C.E. Perkins, pp. 75-138. USA: Addison-Wesley. 2001.
[44] Bettstertter, C. and S. Konig. On the Message and Time Complexity of a Distributed Mobility-Adaptive Clustering Algorithm in Wireless Ad Hoc Networks. In Proc. 4th European Wireless, February 2002, Florence, Italy, pp. 128-134.
[45] Chiang, C.C., H.K. Wu, W. Liu and M. Gerla. Routing in Clustered Multihop, Mobile Wireless Networks with Fading Channel. In Proc. IEEE Singapore International Conference on Networks (SICON), April 1997, Singapore, pp. 197-211.
[46] Baker, D.J. and A. Ephremides. The Architectural Organization of a Mobile Radio Network via a Distributed Algorithm, IEEE Transactions on Communication, pp. 1694-1701. 1981.
[47] Baker, D.J., A. Ephremides and J.A. Flynn. The Design and Simulation of a Mobile Radio Network with Distributed Control, IEEE Journal on Selected Areas in Communications, *Vol. SAC-2*, pp. 226-237. 1984.
[48] Parekh, A.K. Selecting Routers in Ad Hoc Wireless Networks, Journal of Brazilian Computer Society, *1(2)*, pp. 75-86. 1994.
[49] Basagni, S. Distributed Clustering for Ad Hoc Networks. In Proc. ISPAN ’99, June 1999, Perth, Australia, pp. 310-315.
[50] Amis, A.D., R. Prakash, T.H. Vuong and D.T. Huynh. Max-Min *d*-Cluster Formation in Wireless Ad Hoc Networks. In Proc. IEEE INFOCOM 2000, March 2000, Tel-Aviv, Israel, pp. 32-41.
[51] Chen, Y.P. and A.L. Liestman. Approximating Minimum Size Weakly-Connected Dominating Sets for Clustering Mobile Ad Hoc Networks. In Proc. 3rd ACM International Symposium on Mobile Ad Hoc Networking and Computing, ACM Press, June 2002, Lausanne, Switzerland, pp. 165-172.
[52] Basu, P., N. Khan and D.C. Little. Mobility Based Metric for Clustering in Mobile Ad Hoc Networks. In Proc. Workshop on Distributed Computing Systems, 2001, pp. 413-418.
[53] An, B. and S. Papavassiliou. A Mobility-Based Clustering Approach to Support Mobility Management and Multicast Routing in Mobile Ad-hoc Wireless Networks, International Journal of Network Management 11, pp. 387-395. 2001.
[54] Taek, J.K. and M. Gerla. Efficient Flooding with Passive Clustering (PC) in Ad Hoc Networks, ACM SIGCOMM Computer Communication Review, *32(1)*, pp. 44-56. 2002.
[55] Belding-Royer, E.M. Hierarchical Routing in Ad Hoc Mobile Networks, Wireless Communications and Mobile Computing 2002, *2(5)*, pp. 515-532. 2002.

[56] Krishna, P., N. Vaidya, M. Chatterjee, and Pardhan. A Cluster-Based Approach for Routing in Dynamic Networks, ACM SIGCOMM Computer Communication Review, *27(2)*, pp. 49-64. 1997.
[57] Fernandess, Y. and D. Malkhi. *k*-Clustering in Wireless Ad Hoc Networks. In Proc. 2nd ACM International Workshop on Principles of Mobile Computing, October 2002, Toulouse, France, pp. 31-37.
[58] McDonald, A. and T. Znati. A Mobility-Based Framework for Adaptive Clustering in Ad Hoc Networks, IEEE Journal on Selected Areas in Communications, *17(8)*, pp. 1466-1487. 1999.
[59] Clark, B.N., C.J. Colboum and D.S. Johnson. Unit Disk Graphs, Discrete Mathematics, *86*, pp. 165-177. 1990.
[60] Bharghavan, V. and B. Das. Routing in Ad Hoc Networks Using Minimum Connected Dominating Sets. In Proc. Int. Conf. Communication '97, June 1997, Montreal, Canada, pp. 376-380.
[61] Das, B., R. Sivakumar and V. Bharghavan. Routing in Ad-hoc Networks Using a Spine. In Proc. Int Conf. Comput. And Commun. Networks (IC3N), September 1997, Las Vegas, NV, USA, pp. 34.
[62] Sivakumar, R., B. Das, and V. Bharghavan. An Improved Spine-Based Infrastructure for Routing in Ad Hoc Networks. In Proc. IEEE Symp. Comput. And Commun., June 1998, Athens, Greece.
[63] Min, M., F. Wang, D-Z. Du, and P.M. Pardalos. A Reliable Virtual Backbone Scheme in Mobile Ad Hoc Networks. In Proc. 1st IEEE International Conference on Mobile Ad-Hoc and Sensor Systems (MASS), October 2004.
[64] Marathe, M.V., H. Breu, H.B. Hunt III, S.S. Ravi and D.J. Rosenkrantz. Simple Heuristics for Unit Disk Graphs, Networks, *Vol. 25*, pp. 59-68. 1995.
[65] Guha, S. and S. Khuller. Approximation Algorithms for Connected Dominating Sets, Algorithmica, *20(4)*, pp. 374-387. 1998.
[66] Alzoubi, K.M., P. Wan and O. Frieder. Distributed Heuristics for Connected Dominating Sets in Wireless Ad Hoc Networks, Journal of Communications and Networks, *4(1)*, pp. 22-29. 2002.
[67] Cidon, I. and O. Mokryn. Propagation and Leader Election in Multihop Broadcast Environment. In Proc. 12th International Symposium on DIStributed Computing (DISC'98), September 1998, Andros, Greece, pp. 104-119.
[68] Alzoubi, K.M., P. Wan and O. Frieder. Message-Optimal Connected Dominating Sets in Mobile Ad Hoc Networks. In Proc. MobiHOC 2002, June 2002, Switzerland, pp. 157-164.
[69] Wu, J. and H. Li. On Calculating Connected Dominating Set for Efficient Routing in Ad Hoc Wireless Networks. In Proc. 3rd International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications, August 1999, Seattle, USA, pp. 7-14.
[70] Wan, P., K.M. Alzoubi and O. Frieder. Distributed Construction of Connected Dominating Set in Wireless Ad Hoc Networks. In Proc. IEEE INFOCOM 2002, June 2002, New York, pp. 141-149.
[71] Wang, Y., W. Wang and X-Y. Li. Distributed Low-Cost Backbone Formation for Wireless Ad Hoc Networks. In Proc. ACM MobiHoc 2005, March 2005, Urbana-Champaign, IL, USA, pp. 2-13.
[72] Ephremides, A., J. Wieselthier and D. Baker. A Design Concept for Reliable Mobile Radio Network with Frequency Hopping Signaling, Proc. of IEEE 75, pp. 56-73. 1987.
[73] Basagni, S., I. Chlamtac and A. Farag. A Generalized Clustering Algorithm for Peer-to-Peer Networks. In Proc. Workshop on Algorithmic Aspects of Communication, Satellite Workshop of ICALP'97, July 1997, Bologna, Italy.

[74] Chatterjee, M., S.K. Das and D. Turgut. WCA: A Weighted Clustering Algorithm for Mobile Ad Hoc Networks, Cluster Computing 5, Kluwer Academic Publishers, pp. 193-204. 2002.

[75] Basagni, S. Distributed and Mobility-Adaptive Clustering for Multimedia Support in Multihop Wireless Networks. In Proc. IEEE Vehicular Technology Conference, September 1999, Amsterdam, Netherlands, pp. 889-893.

[76] Basagni, S. Finding a Maximal Weighted Independent Set in Wireless Networks, Telecommunication Systems, *18(1-3)*, Kluwer Academic Publishers, pp. 155-168. 2001.

[77] Nocetti, F.G., J.S. Gonzalez and I. Stojmenovic. Connectivity-Based *k*-Hop Clustering in Wireless Networks, Telecommunication Systems, *22(1-4),* Kluwer Academic Publishers, pp. 205-220. 2003.

[78] Banerjee, S. and S. Khuller. A Clustering Scheme for Hierarchical Control in Multihop Wireless Networks. In Proc. IEEE INFOCOM 2001, April 2001, Anchorage, Alaska, pp. 1028-1037.

[79] Camp, T., J. Boleng and V. Davies. A Survey of Mobility Models for Ad Hoc Network Research, Wireless Communication & Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications, *2(5)*, pp. 483-502. 2002.

[80] Jardosh, A., E.M. Belding-Royer, K.C. Almeroth and S. Suri. Towards Realistic Mobility Models for Mobile Ad Hoc Networks. In Proc. MobiCom ’03, September 2003, San Diego, CA, pp. 217-229.

[81] Cain, B., S. Deering, I. Kouvelas, B. Fenner and A. Thyagarajan. Internet Group Membership Protocol Version 3, RFC 3376, The Internet Society (2002). Available from: <http://tools.ietf.org/html/rfc3376>.

[82] Karp, R.M. Reducibility among Combinatorial Problems. In Complexity of Computer Communications, ed by R. Miller and J. Thatcher, pp. 85-103, New York: Plenum Press. 1972.

[83] Robins, G. and A. Zelikovski. Improved Steiner Tree Approximation in Graphs. In Proc. 11th ACM-SIAM Symposium on Discrete Algorithms, January 2000, San Francisco, CA, pp. 770-779.

[84] Agrawal, A., P. Klein and R. Ravi. When Trees Collide: An Approximation Algorithm for the Generalized Steiner Tree Problem on Networks, SIAM Journal on Computing, *24(3)*, pp. 440-456. 1995.

[85] Westbrook, J. and D.K. Yan. Greedy Algorithms for the On-Line Steiner Tree and Generalized Steiner Problems. In Proc. 3rd Workshop on Algorithms and Data Structures, August 1993, Montreal, Canada, pp. 621-633.

[86] Imase, M. and B.M. Waxman. Dynamic Steiner Tree Problem, SIAM Journal on Discrete Mathematics, *4(3)*, pp. 369-384. 1991.

[87] Obraczka, K. and G. Tsudik. Multicast Routing Issues in Ad Hoc Networks. In Proc. of IEEE ICUPC, October 1998, Florence, Italy, pp. 751-756.

[88] Ni, S-Y., Y-C, Tseng, Y-S. Chen and J-P. Sheu. The Broadcast Storm Problem in a Mobile Ad Hoc Network, Wireless Networks, 8(2/3), pp. 153-167. 2002.

[89] Chiang, C.C., M. Gerla and L. Zhang. Forwarding Group Multicast Protocol (FGMP) for Multihop, Mobile Wireless Networks, Baltzer Cluster Computing, *1(2)*, pp. 187-196. 1998.

[90] Jetcheva, J.G., Y. Hu, D.A. Maltz and D.B. Johnson. A Simple Protocol for Multicast and Broadcast in Mobile Ad Hoc Networks. Internet Draft, Internet Engineering Task Force (2001). draft-ietf-manet-simple-mbcast-01.txt.

[91] Toh, C-K., G. Guichala and S. Bunchua. ABAM: On-Demand Associativity-Based Multicast Routing for Ad Hoc Mobile Networks. In Proc. IEEE Vehicular Technology Conference (VTC 2000), September 2000, Long Beach, CA, pp. 987-993.

[92] Toh, C-K. Associativity-Based Routing for Ad-hoc Mobile Networks, Wireless Personal Communications, *4(2)*, pp. 1-36. 1997.

[93] Devarapalli, V. and D. Sidhu. MZR: A Multicast Protocol for Mobile Ad Hoc Networks. In Proc. IEEE International Conference on Communications (ICC), June 2001, Helsinki, Finland, pp. 886-891.
[94] Haas, Z.J., M.R. Pearlman and P. Samar. The Zone Routing Protocol (ZRP) for Ad Hoc Networks. Internet Draft, Internet Engineering Task Force (2002). draft-ietf-manet-zone-zrp-04.txt.
[95] Ji, L. and M.S. Corson. Differential Destination Multicast – A MANET Multicast Routing Protocol for Small Groups. In Proc. INFOCOM 2001, April 2001, Anchorage, Alaska, pp. 1192-1202.
[96] Perkins, C.E. and E.M. Belding-Royer. Ad Hoc On-Demand Distance Vector Routing. In Proc. 2nd IEEE Workshop on Mobile Computing Systems and Applications, February 1999, New Orleans, LA, pp. 90-100.
[97] Belding-Royer, E.M. and C.E. Perkins. Multicast Operation of the Ad-hoc On-Demand Distance Vector Routing Protocol. In Proc. 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking (Mobicom '99), August 1999, Seattle, USA, pp. 207-218.
[98] Perkins, C.E. and E.M. Royer. The Ad Hoc On-Demand Distance-Vector Protocol. In Ad Hoc Networking, ed by C.E. Perkins, pp. 173-219. USA: Addison-Wesley. 2001.
[99] Bommaiah, E., M. Liu, A. McAuley and R. Talpade. AMRoute: Adhoc Multicast Routing Protocol. Internet Draft, Internet Engineering Task Force (1998). draft-talpade-manet-amroute-00.txt.
[100] Xie, J., R.R. Talpade, A. Mcauley and M. Liu. AMRoute: Ad hoc Multicast Routing Protocol, Mobile Networks and Applications, *7(6)*, pp. 429-439. 2002.
[101] Lee, S.J., W. Su, J. Hsu, M. Gerla and R. Bagrodia. A Performance Comparison Study of Ad Hoc Wireless Multicast Protocols. In Proc. IEEE INFOCOM 2000, March 2000, Tel-Aviv, Israel, pp. 565-574.
[102] Wu, C.W. and Y.C. Tay. AMRIS: A Multicast Protocol for Ad hoc Wireless Networks. In Proc. IEEE MILCOM '99, November 1999, Atlantic City, NJ, pp. 25-29.
[103] Ji, L. and M.S. Corson. A Lightweight Adaptive Multicast Algorithm. In Proc. IEEE GLOBECOM '98, December 1998, Sydney, Australia, pp. 1036-1042.
[104] Park, V.D. and M.S. Corson. A Highly Adaptive Distributed Routing Protocol for Mobile Wireless Networks. In Proc. IEEE INFOCOM 1997, April 1997, Kobe, Japan, pp. 1405-1413.
[105] Meggers, J. and G. Filios. Multicast Communication in Ad Hoc Networks. In Proc. IEEE 48th Vehicular Technology Conference, May 1998, Ottawa, Canada, pp. 372-376.
[106] Johnson, D.B., D.A. Maltz, Y. Hu and J. G. Jetcheva. The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks. Internet Draft, Internet Engineering Task Force (2002). draft-ietf-manet-dsr-07.txt.
[107] Lee, S.J., M. Gerla and C.C. Chiang. On-Demand Multicast Routing Protocol. In Proc. IEEE WCNC '99, September 1999, New Orleans, LA, pp. 1298-1304.
[108] Garcia-Luna-Aceves, J.J. and E.L. Madruga. A Multicast Routing Protocol for Ad-Hoc Networks. In Proc. IEEE INFOCOM 1999, March 1999, New York, pp. 1380-1394.
[109] Kang, H. and M. Lee. A Multi-Source Multicast Routing Protocol for Ad-Hoc Networks. In Proc. ICOIN 2002, Cheju Island, Korea, pp. 309-320.
[110] Lee, S. and C. Kim. Neighbor Supporting Ad hoc Multicast Routing Protocol. In Proc. 1st Annual Workshop on Mobile Ad Hoc Network & Computing (MOBIHOC) 2000, August 2000, Boston, MA, pp. 37-44.
[111] Das, S.K., B.S. Manoj and C.S.R. Murthy. Dynamic Core Based Multicast Routing Protocol. In Proc. ACM Mobihoc 2002, June 2002, Lausanne, Switzerland, pp. 24-35.
[112] Sinha, P., R. Sivakumar and V. Bharghavan. CEDAR: Core Extraction Distributed Ad hoc Routing. In Proc. IEEE INFOCOM 1999, March 1999, New York, pp. 202-209.

[113] Gerla, M., X. Hong and G. Pei. LANMAR: Landmark Routing for Large Scale Wireless Ad Hoc Networks with Group Mobility. In Proc. of IEEE/ACM MobiHoc, August 2000, Boston, MA, pp. 11-18.
[114] Jetcheva, J.G. and D.B. Johnson. Adaptive Demand-Driven Multicast Routing in Multihop Wireless Ad hoc Networks. In Proc. 2001 ACM International Symposium on Mobile Ad Hoc Networking & Computing, October 2001, Long Beach, CA, pp. 33-44.
[115] Gui, C. and P. Mohapatra. Scalable Multicasting in Mobile Ad Hoc Networks. In Proc. IEEE INFOCOM 2004, March 2004, Hong Kong, pp..
[116] Imielinski, T. and J.C. Navas. Geographic Addressing, Routing and Resource Discovery with the Global Positioning System, Communications of the ACM Journal, 1997.
[117] Ko, Y.B. and N.H. Vaidya. Geocasting in Mobile Ad Hoc Networks: Location-Based Multicast Algorithms. In Proc. IEEE WMCSA, February 1999, New Orleans, LA, pp. 101-110.
[118] Ko, Y.B. and N.H. Vaidya. GeoTORA: A Protocol for Geocasting in Mobile Ad Hoc Networks. In Proc. IEEE ICNP 2000, November 2000, Osaka, Japan, pp. 240-250.
[119] Liao, W-H., Y.C. Tseng and J.P. Sheu. GeoGRID: A Geocasting Protocol for Mobile Ad Hoc Networks Based on GRID, Journal of Internet Technology, *1(2)*, pp. 23-32. 2000.
[120] Liao, W-H., Y.C. Tseng and J.P. Sheu. GRID: A Fully Location-Aware Routing Protocol for Mobile Ad Hoc Networks. Telecommunication Systems, *18(1-3)*, pp. 37-60. 2001.
[121] Viswanath, K., K. Obraczka and G. Tsudik. Exploring Mesh and Tree Based Multicast Routing Protocols for MANETs, IEEE Transactions on Mobile Computing, *5(1)*, pp. 28-42. 2006.
[122] Lee, S.J., W. Su and M. Gerla. On-Demand Multicast Routing Protocol in Multihop Wireless Mobile Networks, ACM/Kluwer Mobile Networks and Applications, special issues on Multipoint Communications in Wireless Mobile Networks*, 7(6)*, pp. 441-453. 2002.
[123] Garcia-Luna-Aceves, J. J. and E.L. Madruga. The Core-Assisted Mesh Protocol, IEEE Journal on Selected Areas in Communications, *17(8)*, pp. 1380-1394. 1999.
[124] Gupta, S.K.S. and P.K. Srimani. An Adaptive Protocol for Reliable Multicast in Multihop Radio Networks. In Proc. 2nd IEEE Workshop on Mobile Computing Systems and Applications (WMCSA'99), February 1999, New Orleans LA, pp. 111-122.
[125] Jaikaeo, C. and C. Shen. Adaptive Backbone-based Multicast for Ad Hoc Networks, In Proc. IEEE International Conference on Communications (ICC 2002), April 2002, New York.
[126] Hong, X., M. Gerla, G. Pei and C. Chiang. A Group Mobility Model for Ad Hoc Wireless Networks. In Proc. ACM/IEEE MSWiM, August 1999, Seattle, WA, pp. 53-60.
[127] Huang, R., G. V. Záruba. Location Tracking in Mobile Ad Hoc Networks Using Particle Filters. In Proc. of the 4th International Conference on Ad Hoc networks and Wireless (Ad Hoc NOW), vol. 3738, October 2005, Cancun, Mexico, pp. 85-98.
[128] Castaneda, R. and S.R. Das. Query Localization Techniques for On-Demand Routing Protocols in Ad Hoc Networks. In Proc. 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking**,** August 1999, Seattle, WA, pp. 186-194.
[129] Iwata, A., C.-C. Chiang, G. Pei, M. Gerla and T.-W. Chen. Scalable Routing Strategies for Ad hoc Wireless Networks, IEEE Journal on Selected Areas in Communications, Special Issue on Ad hoc Networks, pp. 1369-1379. 1999.
[130] Jiang, M., J. Li and Y. C. Tay. Cluster Based Routing Protocol (CBRP), Internet Draft, Internet Engineering Task Force (1999). draft-ietf-manet-cbrp-spec-01.txt.
[131] Belding-Royer, E.M. Multi-Level Hierarchies for Scalable Ad Hoc Routing. Wireless Networks, *9(5)*, pp. 461-478. 2003.
[132] Bettstetter, C. and R. Krausser. Scenario-Based Stability Analysis of the Distributed Mobility-Adaptive Clustering (DMAC). In Proc. ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHOC), October 2001, Long Beach, CA, pp. 232-241.

[133] Xu, K. and M. Gerla. A Heterogeneous Routing Protocol Based on a New Stable Clustering Scheme. In Proc. IEEE Military Communications Conferences (MILCOM' 02), October 2002, Anaheim, CA, pp. 838-843.
[134] Liang, B. and Z.J. Haas. Virtual Backbone Generation and Maintenance in Ad Hoc Network Mobility Management. In Proc. 19th IEEE INFOCOM 2000, March 2000, Tel-Aviv, Israel, pp. 1293-1302.
[135] Broch, J., D.A. Maltz, D.B. Johnson, Y.-C. Hu and J. Jetcheva. A Performance Comparison of Multihop Wireless Ad Hoc Network Routing Protocols. In Proc. 4th Annual ACM/IEEE International Conference on Mobile Computing and Networking, October 1998, Dallas, TX, pp. 85-97.
[136] Waal, C. and M. Gerharz. BonnMotion: A Mobility Scenario Generation and Analysis Tool. Communication Systems Group, Institute of Computer Science IV, University of Bonn, Germany (2003). Cited by 09 August 2006. Available from: <http://web.informatik.uni-bonn.de/IV/Mitarbeiter/dewaal/BonnMotion/>.
[137] Sucec, J. and I. Marsic. Hierarchical Routing Overhead in Mobile Ad Hoc Networks, IEEE Transactions on Mobile Computing, *3(1)*, pp. 46-56. 2004.
[138] Clausen, T., C. Dearlove and J. Dean. MANET Neighborhood Discovery Protocol (NHDP), Internat Draft, Internet Engineering Task Force (2006). draft-ietf-manet-nhdp-00.
[139] Perkins, C.E., E.M. Belding-Royer, S.R. Das. Ad hoc On-Demand Distance Vector (AODV) Routing, Internet Draft, Internet Engineering Task Force (2002). draft-ietf-manet-aodv-10.txt.
[140] Wieselthier, J.E., G.D. Nguyen and A. Ephremides. On the Construction of Energy-Efficient Broadcast and Multicast Trees in Wireless Networks. In Proc. IEEE INFOCOM 2000, March 2000, Tel-Aviv, Israel, pp. 584-594.
[141] Law, L.K., S.V. Krishnamurthy and M. Faloutsos. On Evaluating the Trade-offs between Broadcasting and Multicasting in Ad Hoc Networks. In Proc. of the IEEE MILCOM, November 2004, Monterey, CA.
[142] Akyildiz, I.F., X. Wang and W. Wang. Wireless Mesh Networks: A Survey, Computer Networks Journal (Elsevier), *47(4)*, pp. 445-487. 2005.
[143] Xue, M.Q., I.I. Er, and W.K.G. Seah. Analysis of Clustering and Routing Overhead for Clustered Mobile Ad Hoc Networks. In Proc. of the 26th International Conference on Distributed Computing Systems (ICDCS 2006), July 2006, Lisboa, Portugal.

APPENDIX A

GRAPE PACKET FORMATS

A.1 Multicast Join Request Packet (MREQ)

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Type      |  Dist TO NP    |  Time To Live |   Fwd Count   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Multicast Group IP Address                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Sequence Number                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Source IP Address                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   Previous Hop IP Address                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Nearest Participant IP Address                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type

01; GRAPE Multicast Join Query (MREQ).

Dist To NP

Number of hops away from nearest participant(Used by BODS algorithm)

Time To Live

Number of hops this packet can traverse.

Fwd Count

The number of hops traveled so far by this packet.

Multicast Group IP Address

The IP address of the multicast group.

Sequence Number

The sequence number assigned by the source to uniquely identify the packet.

Source IP Address

The IP address of the node originating the packet.

Previous Hop IP Address

The IP address of the last node that has processed this packet.

Nearest Participant IP Address

The IP address of the nearest participant (used by BODS algorithm)

A.2 Multicast Join Reply Packet (MREP)

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Type     |    Hop Count  |   Resend Flag |   Reserved     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Multicast Group IP Address                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   Previous Hop IP Address                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Sequence Number                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Multicast Destination IP Address                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Primary Previous Hop IP Address                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Secondary Previous Hop IP Address                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type

02; GRAPE Multicast Join Reply (MREP).

Hop Count

The number of hops traveled so far by this packet.

Resend Flag

The flag that will be turn on if the secondary path should be in use.

Reserved

Sent as 0; ignored on reception.

Multicast Group IP address

The IP address of the multicast group.

Previous Hop IP Address

The IP address of the last node that has processed this packet.

Sequence Number

The sequence number assigned by the previous hop node to uniquely identify the packet.

Primary Previous Hop IP Address

The IP address of the primary next node that this packet is targeted to.

Secondary Previous Hop IP Address

The IP address of the secondary next node that this packet is targeted to.

A.3 Multicast Member Join/Leave Packet (MemberJL)

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Type       |   Hop Count  |    Join Flag  |   Reserved     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   Multicast Group IP Address                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Sequence Number                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                Multicast Destination IP Address                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Next Hop IP Address                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type

03; GRAPE Multicast Member Join or Leave (MemberJL).

Hop Count

The number of hops traveled so far by this packet.

Join Flag

The flag that will be turn on if the packet is for joining the group.

Reserved

Sent as 0; ignored on reception.

Multicast Group IP address

The IP address of the multicast group to which the packet initiator intends to join.

Sequence Number

The sequence number assigned by the previous hop node to uniquely identify the packet.

Next Hop IP Address

The IP address of the next node that this packet is targeted to, which is also the parent node of the packet initiator.

APPENDIX B

GRAPE DATA STRUCTURES

B.1 Format of MREQ-Cache Entry

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Multicast Group IP Address                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Sequence Number                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  MREQ Initiator IP Address                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Forward Count                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Nearest Participant IP Address                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Distance to Nearest Participant                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-|
|                   Previous Hop IP Address                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Multicast Group IP address

The IP address of the multicast group.

Sequence Number

The sequence number of the received MREQ packet.

MREQ Initiator IP Address

The IP address of the source node that initiates this MREQ packet.

Forward Count

The hop count that this MREQ packet has traveled so far.

Nearest Participant IP Address

The IP address of the nearest participant (used by BODS algorithm)

Distance to Nearest Participant

Number of hops away from the nearest participant (used by BODS algorithm)

Previous Hop IP Address

The IP address of the last node that has processed this packet.

B.2 Format of MG-Flag-Cache Entry

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |                  Multicast Group IP Address                     |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |                        Forward Flag                             |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Multicast Group IP address

The IP address of the multicast group.

Forward Flag

The Boolean flag that will be turn on if the node is a forwarding node.