ARCHITECTURE AND PROTOCOLS FOR SUPPORTING ROUTING AND QUALITY-OF-SERVICE IN MOBILE AD HOC NETWORKS

Navid Nikaein

Pour Obtenir le Grade de

Docteur
de l'École Polytechnique Fédérale de Lausanne (EPFL)

Date de Soutenance: 20 Juin 2003
Composition du Jury:

Prof. Bixio Rimoldi	Président
Prof. Jean-Yves Le boudec	Rapporteur
Prof. Philippe Jacquet	Rapporteur
Prof. Rahim Tafazolli	Rapporteur
Prof. Jon Crowcroft	Rapporteur
Prof. Refik Molva	Rapporteur
Prof. Christian Bonnet	Directeur de Thèse

Thése réalisé au sein de l'Institut Eurecom

AUTHOR'S ADDRESS:
Navid Nikaein
Institut Eurecom
Departement de Communication Mobile
2229, route des Crêtes
B.P. 193
06904 Sophia-Antipolis – France
Email: Navid.Nikaein@eurecom.fr
URL: http://www.eurecom.fr/~nikaeinn

Acknowledgments

I would like to express my most sincere thanks to my supervisor, Professor Christian Bonnet, who gave me the opportunity to work on interesting problems at my pace. His constant encouragement has been invaluable. I am grateful to him for his technical insights, constructive criticism and friendship. I feel very glad and lucky for the chance that Professor Christian Bonnet and Eur´ecom gave to me.

I further wish to thank Professor Rahim Tafazolli from the CCSR, Surrey University, England, with whom I had many stimulating discussions during my visit at CCSR. This collaboration was a pleasure which I hope will continue in the future.

I am also thankful to all the jury members of my thesis for their comments which helped me to improve the quality of my thesis.

Thanks to the other coauthors and people I have worked with during these years: Paolo Penna, Neda Nikaein, Sergio Loureiro, Shiyi Wu, Pietro Michiardi, Nadeem Akhtar, and Ashish Rastogi.

I am grateful to all of my friends that during these years helped me to develop myself as a person: Neda Nikaein, Amir-Hossein Falah, Payam Talebi, Afshin Aman-Zadeh, Sergio Loureiro, Paolo Penna, Alejandro Ribes Cortes, Nilofar Emami-Shahri, David Turner, Maxime Guillaud, David Mary, Nicolas de Saint Aubert, and Carine Aud´e.

My special gratitude goes to my mother Parvin, my father Hashem, and my sister Neda for their love, support and encouragement.

I dedicate this thesis to my parents and to my supervisor

Summary

An architecture that separates topology management from route determination is proposed for quality of service routing in mobile ad hoc network. In this architecture, topology management adjusts the network topology as nodes move and environment changes; while route determination generates and selects path(s), and forwards user traffic between source and destination. This architecture optimizes network and routing performances according to two criteria: network quality and application requirements. Topology management forms a logical structure between nodes with respect to the network quality in order to optimize network performance. Route determination, on the other hand, finds the most suitable path on the top of the logical structure according to the application requirements so as to optimize routing performance.

CONTENTS

I Introduction **5**

A Motivations and Objectives 6

B Outline of Thesis 9

II Related Work **11**

A The Design Challenges of Ad Hoc Routing 12

B Conventional Routing Protocols 14

C Current Ad Hoc Routing Protocols 14

D Discussion 15

E Current IETF Routing Protocols 16

F Contributions 17

F.1 Topology Management 17

F.2 Route Determination 18

F.3 Quality of Service Support 19

F.4 Summary 19

G Comparison 20

III Topology Management in Mobile Ad Hoc Networks **21**

A Motivation 22

B Basic Idea 23

C Preliminary Definitions 24

D DDR – Distributed Dynamic Routing Algorithm 26

D.1 Preferred Neighbor Election 27

D.2 Forest Construction 29

D.3 Intra-zone Clustering 30

D.4 Inter-zone Clustering 32

D.5 Zone Naming 33
D.6 Zone Partitioning 35
D.7 Zone Maintenance 35
D.8 Summary 36
E Correctness 36
F Pseudo Code and Flow Chart 37
G Mathematical Analysis 38
H New Criteria for Preferred Neighbor Election 42
H.1 Quality of Connectivity at the Network Layer 42
H.2 Forest of High Quality Nodes 44
I Behavioral Results on Topology Management Strategy 44
J Protocol Model 48
K Simulation Model 49
L Performance Results 50
L.1 Packet Delivery Fraction 50
L.2 Average end-to-end delay of data packets 50
L.3 Routing overhead 52
M Architecture 52
N Conclusion 57

IV HARP—Hybrid Ad Hoc Routing Protocol 59

A Routing Procedures 61
A.1 Advance Routing 62
A.2 Intra-zone Routing 62
A.3 Inter-zone Routing 64
A.4 Path Maintenance 67
A.5 Query Localization Mechanism 68

A.6 Summary 71
B Simulation Model 71
C Performance Evaluation 73
C.1 Packet Delivery Fraction 73
C.2 Average end-to-end delay of data packets 74
C.3 Routing overhead 77
C.4 Discussion 77
C.5 Related Work 82
D Conclusion 83

V Quality of Service Support 85
A Background on Quality-of-Service Models 87
B Proposed Quality-of-Service Model 88
B.1 Assumption 88
B.2 Intuition 89
B.3 Network Layer Metrics 90
B.4 Quality of Service Classes and Metrics' Mapping 91
B.5 Application Metrics 92
C Performance Evaluation 93
D Additional Mechanisms 95
D.1 Path Stability Period 95
D.2 Service Differentiation in Nodes 95
E Extensions for Routing Protocols 97
E.1 Proactive Approach 97
E.2 Reactive Approach 98
F Conclusion and Future Work 99

VI Conclusion 101

Bibliography **104**

Glossary **111**

List of Tabless **114**

List of Figures **115**

Curriculum Vitae **118**

List of Publication **120**

CHAPTER I

Introduction

THE growth of wireless communications coupled with high-speed broadband technology has led to a new era in telecommunications. In third generation mobile networks, efforts are undertaken to merge many technologies and systems to support a wide range of traffic types with various quality-of-service requirements. As wireless communication channels are highly affected by unpredictable temporal/spatial factors like co-channel interference, adjacent channel interference, propagation path loss and multipath fading; it is necessary to embed various adaptive mechanisms making these systems self-adaptive and more efficient, satisfying application best requirements and mitigating bad effects of wireless channels. Broadly, there are two major architectures for wireless networking: *single-hop* and *multi-hop*. The single-hop model is based on the cellular model, provides one-hop wireless connectivity between mobile hosts and static nodes known as *base station*. This type of networks relies on a fixed backbone infrastructure that interconnects all base stations by means of high-speed wired links. Typically, a certain number of base stations are located on a geographical region in order to obtain a coverage of such region, so whatever the position of the user(s) is, there always exists a base station that connects that user to the network. The multi-hop model, on the other hand, requires neither a fixed and/or wired infrastructure nor a predefined interconnectivity. One of the most popular type of multi-hop networks is called *mobile ad hoc networks* (Manet) [1]. They consist of a collection of wireless mobile nodes forming dynamic autonomous networks through a fully mobile infrastructure. This means that nodes communicate with each other without the intervention of centralized access points or base stations, and hence they are not relying on any fixed infrastructure. In such networks, each node acts as a host, and may act as a router if it volunteers to carry traffic. Combined with the lack of fixed infrastructure, the limited

transmission range of wireless network interfaces indicates the need for *multihop* routing (i.e. store-and-forward packets) to exchange data between nodes in the networks [2]. This is why the literature sometimes uses the term *multihop* networks for Manets. Indeed, the merits of having multihop networks were discovered in the early 1970s, and were first referred to as *packet radio networks* [3]. Recognizing the advantages of packet switching in a mobile wireless environment, research efforts were initiated to develop packet radio networks providing an efficient means of sharing a broadcast radio channel as well as coping with changing and incomplete connectivity [4]. Of course, Manet will neither be a replacement nor an alternative to current and future infrastructure-based networks. Instead, they will complement these infrastructures in cases where cost, constraints, or environment require infrastructure-less solutions [5].

One of the original motivation for Manet is found in the military need for battlefield survivability [6]. Indeed, a fully mobile platform provides a fluid network where each entity moves about freely without any of restrictions imposed by a fixed platform. Also, the military cannot rely on access to a fixed, pre-placed communication infrastructure in battlefield environment. In some regions, such as the desert or in space, there is no terrestrial communication infrastructure. In other regions, access is unavailable or unreliable because of the destruction of the local infrastructure or eavesdropping of the information. Therefore, a rapidly and easily deployable mobile infrastructure seems very promising in such situations. Another motivation factor is derived from the inability of frequencies much higher than 100 MHz to propagate beyond line of sight (LOS). Terrain and man-made obstacles can also prevent LOS connectivity. Thus, multihop routing is required to exchange messages between users that are not within LOS of each other.

Manet are viewed as suitable systems which can support some specific applications such as *personal area network* (PAN) and *group area network* (GAN). PAN enables the communication associated with a single person through different type of devices such as cell phones, laptops, PDA, watch, and etc; as well as virtual reality devices. These devices need to communicate with each other while interacting with their users' activities. The idea of GAN is to create a network between a group of users and/or devices. For instance, communication set-up in exhibitions, conferences, presentations, meetings, lectures, and home; or in military, emergency, and discovery situations.

Many critical issues need to be addressed in Manet including routing (unicasting, multicasting, and broadcasting), mobility management, quality of service support, radio interface, power management, security. This dissertation tackles the problem of topology management, unicast routing, and quality-of-service (QoS) support for mobile ad hoc networks.

A MOTIVATIONS AND OBJECTIVES

Mobile ad hoc networking Manet is a challenging task due to frequent changes in network topology as well as the wireless nature of the network interface. The frequent

changes in network topology implies a limited lifetime for the topology information as a function of mobility rate. Such information has to be periodically updated in order to remain valid. Thus, the more frequent the information becomes updated, the better the node mobility may be managed. On the other hand, wireless nature of the interface implies the limited bandwidth capacity in the network. This interface experiences high collision probability due to its broadcasting nature, as well as high bit error rate in a radio transmission. Therefore, mobile and wireless environments exhibit opposite requirements. Indeed, the mobility requires more information to be send to keep track of the network changes, while the wireless medium has low capacity and hence cannot be used for additional control traffic that is needed to continually update stale information. Furthermore, there exists some critical factors and parameters that affect the overall performance of mobile ad hoc networks including: network size and connectivity, mobility models, mobility rate, traffic patterns, traffic load, and terrestrial limitations [6]. These factors and parameters along with inherent characteristics of wireless mobile ad hoc networks may result in unpredictable variations in the network performance. Recently, Gupta and Kumar demonstrated that for n fixed nodes with random traffic pattern forming a wireless network, the per-node capacity is $\Theta(1/\sqrt{n \log n})$ [7]. This means that the throughput per node decreases at $1/\sqrt{(n)}$. Therefore in a wireless environment, the total efficiency benefits from the number of nodes, but the performance per node decreases with the number of nodes. Jinyang Li et al. examined the effect of different traffic patterns on per node capacity. They showed that random choice of destination causes a tendency for more packets to be routed through the center of the network than along the edges, which limits the capacity of the network. The authors concluded that the less local the traffic pattern is, the faster per node capacity degrade with the size of network. Grossglauser and Tse studied the effect of mobility on the capacity of ad hoc networks, and showed that capacity improvement can be achieved in the presence of mobility [8]. Camp et al. have provided valuable simulation results that demonstrate the performance of an ad hoc network protocol can vary significantly with different mobility models [9]. They also observed that the choice of mobility model may require a specific traffic pattern which significantly influence protocol performance. D. D. Perkins et al. evaluated the impact of mobility rate, network size, and traffic load on the protocol performance [10]. They observed that the *number* of traffic sources has the strongest impact on the protocol performance followed by the mobility rate and network size. Also, they noticed that increasing the *rate* of traffic load and increasing the number of traffic sources may not produce the same performance results.

Routing protocols for mobile ad hoc networks have the twin design goals of low route delay and low control overhead. The former is important because it indicates the end-to-end delay experienced by data packets. The control overhead is very critical as far as efficient utilization of network resources is concerned. In the ad hoc networking scenario, it is difficult to minimize both the delay and the overhead simultaneously and therefore, *some degree of trade-off is always required*. Proactive routing has low route acquisition delay but significantly large control signaling is needed to maintain exhaustive routing tables. Reactive protocols try to reduce control overhead by discovering routes on-demand at the expense of an increase in route acquisition delay.

Our primary *goal* is to provide an optimal trade-off between overhead and end-to-end delay of a routing protocol while maximizing the network resource utilization. Furthermore, mobile ad hoc networks experience link failure more often. This is because first wireless mobile ad hoc networks potentially have less resources than wired networks and second mobility together with multihop routing may result in link failure, which in turn in a broken path. Hence, a routing protocol that supports quality of service for mobile ad hoc networks should consider the reasons for link failure to improve its performance. Link failure stems from node mobility and lack of network resources. Therefore it is essential to capture the aforesaid characteristics to identify the *quality of connectivity* (QoC) between nodes. Our next *goal* is to identify the quality of connectivity and use it to improve the overall routing performance. To achieve these goals, a new *architecture* that separates topology management from route determination is proposed. In this architecture, topology management adjusts the network topology as nodes move and environment changes; while route determination generates and selects path(s), and forwards user traffic between source and destination. This architecture optimizes network and routing performances according to two criteria: network quality and application requirements. Topology management forms a logical structure between nodes with respect to the network quality in order to optimize network performance. Route determination, on the other hand, finds the most suitable path on the top of the logical structure according to the application requirements so as to optimize routing performance. It is demonstrated that this architecture achieves an optimal trade-off between routing overhead and delay experienced by routing protocols, and it does improve routing performance. Furthermore, this architecture succeeds to provide load balancing in the network.

With the introduction of real-time audio and video applications, specifically two-way voice communications (i.e. telephony) into mobile ad hoc networks, the communication path that is selected between the nodes has to meet additional constraints (i.e. delay). In addition to the destination node, the application must also supply the constraint parameters (i.e. its QoS parameters) to the routing protocol so that a suitable path can be found. The routing protocols that support QoS must be *adaptive* to cope with the time-varying topology and time-varying network resources. For instance, it is possible that a route that was earlier found to meet certain QoS requirements no longer does so due to the dynamic nature of the topology. In such a case, it is important that the network intelligently adapts the session to its new and changed conditions. Many of the candidate protocols establish an end-to-end connection between network applications regardless of the *quality* between intermediate nodes. Thus, for these applications it would be difficult to determine whether any particular application bandwidth or delay requirements can be supported by the communication path. We believe that the quality of service that an application requires depends strictly on the quality of network. Therefore, our last but not least *goal* is to propose a cross-layer quality of service model, which takes into account the quality of the communication path for the path selection procedure. To achieve this goal, we suggest network metrics and application metrics as the additional constraints to the conventional ones to determine paths between source and destination. Network metrics avoid unbalanced network utilization while minimizing the resource consumption in the network; while application

metrics selects a path to meet application requirements. The simulation results will show that the QoS routing improves significantly the delay performance in various network conditions, while keeping the same and sometime better packet delivery fraction. The results confirm that first the model takes the advantage of edge routes in addition to core routes in the network, and second considers the quality of the communication path for supporting application requirements.

B OUTLINE OF THESIS

This thesis focuses on the problem of routing with quality of service support for mobile ad hoc networks. Routing is divided into two sub problems: topology management and route determination, where both of them supports quality of service.

Chapter II provides an overview of some routing protocols proposed for Manet. The major design challenges of a routing protocol are also described. We also present our contribution and compare it with the current IETF routing protocols, namely OLSR, TBRPF, DSR, and AODV.

Chapter III describes the topology management strategy. We introduce the concept of quality of connectivity over time so as to adapt our algorithm to the quality of network. We characterize the behavior of the algorithm under various network density. We demonstrate that the topology management strategy significantly improves routing performance under various traffic load and mobility rate.

In chapter IV, we describe the design of our hybrid ad hoc routing protocol (HARP) and its interaction with topology management strategy, and provide its performance comparison under various traffic load and mobility rate. We demonstrate that our routing protocol together with the topology management strategy succeeds to achieve load balancing in the network.

Chapter V presents the proposed cross-layer quality of service model, which considers the quality of communication path during routing process. We evaluate the performance of our QoS routing with the shortest path routing. The simulation results show that the proposed QoS scheme considerably improves the routing performance, especially in terms of delay, over the standard shortest path routing scheme.

Finally in chapter VI, we draw concluding remarks and summaries of our work and contribution. We discuss the general lessons learned about routing in mobile ad hoc networks, and outline some directions for future research.

CHAPTER II

Related Work

SINCE the advent of Defense Advanced Research Projects Agency (DARPA) packet radio networks in the early 1970s [3], numerous routing protocols have been developed for ad hoc mobile networks. Such protocols have to deal with some important factors as the following:

- ***frequent changes in the network topology—*** since nodes are free to move in an arbitrary manner, the network topology may change randomly and rapidly over time. This causes the network resources to become time varying. As a result, such networks experience link failure more often.
- ***wireless communication—*** which implies limited bandwidth capacity in comparison with wired communication, and differs by the fact that electromagnetic waves propagate in free air of instead if inside cables. Many issues emerge from this fact such as multipath, pathloss, attenuation, shadowing, noise and interference on the channel, making the radio channel a hostile medium whose behavior is difficult to predict [11]. Therefore, wireless communication potentially has low capacity, high collision probability, and high bit error rate.
- ***lack of fixed infrastructure—*** there is no fixed backbone infrastructure for network protocols. Therefore, mobile nodes become the potential network infrastructure and they must act cooperatively to handle network functions.
- ***limited nodes' resources—*** which includes energy, processing capacity, and memory, which are relatively abundant in the wired networks, but may be limited in ad hoc networks and must be preserved. For instance, limited power of the

mobile nodes and lack of fixed infrastructure restrict the transmission range, and create the need for effective multihop routing in mobile ad hoc networks.

These factors and their associated challenges make the protocol design for such environment a very difficult task in the mobile ad hoc networks.

A THE DESIGN CHALLENGES OF AD HOC ROUTING

Routing is defined as a mechanism by which user traffic is directed and transported through the network from the source node to the destination node [12]. The primary goal of routing for mobile ad hoc network is correct and efficient route establishment between nodes. Route construction should be done with a minimum of overhead and bandwidth consumption. It is desirable that a routing protocol to be loop-free, distributed, adaptive, and dynamic. The main routing functionalities are listed below:

1. ***Path generation***— which generates paths according to the assembled and distributed state information of the network and the application. Note that, proactive protocols mainly use link-state or distance-vector algorithm for the path generation; while the reactive protocols apply route discovery procedure.

2. ***Path selection***— which selects appropriate paths based on network and application state information. Path selection procedure in proactive protocols is done based on shortest path algorithms, such as Dijkstra [13] algorithm for the link-state routing and Distributed Bellman-Ford (DBF) [14] for the distance-vector. Reactive protocols basically compute a *metric* such as number of hops during the route discovery procedure and select the path whose metric is the best.

3. ***Data forwarding***— which forwards user traffic along the selected path.

Designing a new routing algorithm may necessitate examining the main strengths and weaknesses of each approach and comparing the different existing approaches [15, 16, 17, 18]. In the literature related to routing protocols used in mobile ad hoc networks, there exist two main design choices: (i) proactive vs. reactive vs. hybrid strategy; (ii) flat vs. hierarchical network architecture.

- ***proactive vs. reactive vs. hybrid strategy***— One of the most critical design challenges for routing in ad hoc networks concerns whether nodes should keep track of routes to all possible destinations, or instead keep track of only those destinations that are of immediate interest. A node in an ad hoc network does not need a route to a destination until that destination is to be the recipient of packets sent by the node, either as the actual source of the packet or as an intermediate node along a path from the source to the destination [2]. Protocols that keep track of

routes for all destinations in the ad hoc networks have the advantage that communications with arbitrary destinations experience minimal initial delay from the point of view of the application. When the application starts, a route can be immediately selected from the routing table. Such protocols are called *proactive* because they store route information even before it is needed. They are also called table driven because routes are available as part of a well-maintained table. These protocols are based on either *link-state* or *distance-vector* algorithm [19, 12]. In this strategy, the trade-off is made between the cost of *full-update* against *no-search* strategies. To overcome the wasted work in maintaining unrequired routes, *on-demand*, or *reactive* protocols have been designed. In these protocols, routing information is acquired only when it is actually needed. Reactive routing protocols save the overhead of maintaining unused routes at each node, but the latency for many applications will drastically increase. Most applications are likely to suffer a long delay when they start because a route to the destination will have to be acquired before the communication can begin. In this approach, *full-search* strategy is used at the expense of *no-update* strategy. Hybrid routing protocols aggregate a set of nodes into zones in the network topology. Then, the network is partitioned into zones and proactive approach is used within each zone to maintain routing information. To route packets between different zones, the reactive approach is used. Consequently, in hybrid schemes, a route to a destination that is in the same zone is established without delay, while a route discovery and a route maintenance procedure is required for destinations that are in other zones. As a result, there is trade-off between the search and update strategies. This trade-off is subject to the size of a zone and the dynamics of a zone.

- ***flat vs. hierarchical architecture*—** Another important design challenge in ad hoc routing concerns whether all nodes should have a uniform responsibility in the network or not. In flat architecture, all nodes carry the same responsibility. Flat architectures do not optimize bandwidth resource utilization in large networks because control messages have to be transmitted globally throughout the network, but they are appropriate for highly dynamic network topology. The scalability decreases significantly when the number of nodes increases. On the contrary, in hierarchical architectures, aggregating nodes into clusters and clusters into super-clusters conceals the details of the network topology. Some nodes, such as cluster heads and gateway nodes have a higher computation communication load than other nodes. Hence, the mobility management becomes complex. The network reliability may also be affected due to single points of failure associated with the defined critical nodes. However, control messages may only have to be propagated within a cluster. Thus, the multilevel hierarchy reduces the storage requirement and the communication overhead of large wireless networks by providing a mechanism for localizing each node. In addition, hierarchical architectures are more suitable for low mobility case. To summarize, flat architectures are more flexible and simpler than hierarchical, but hierarchical architectures are more scalable.

B CONVENTIONAL ROUTING PROTOCOLS

There exists two main approaches: *link state* and *distance vector* [19]. OSPF is a link state routing protocol used for the wired Internet [20], which employs shortest path algorithms such as Dijkstra to compute routes as needed between source and destination for which the link state information is available [13][12]. Experiments have shown that OSPF can consume a very substantial percentage of the available bandwidth in order to maintain the link information [21]. Routing in multihop packet radio networks was based on shortest path routing algorithm [6], such as the *Distributed Bellman-Ford* (DBF) routing algorithm [14, 22]. DBF algorithms are also known as distance vector algorithm because the route table entry for a destination contains a metric, which is often the distance from the node to the destination and the next hop for this destination. These algorithm are easy to program and requires much less storage space compared to link state algorithm. However, they can suffer from very slow convergence (count to infinity problem), and also from short-lived and long-lived loops [23]. Because mobile ad hoc networks constitute a distributed multi-hop network characterized by a time-varying topology and resources, limited bandwidth and limited power, conventional routing protocols may not be appropriate to use.

C CURRENT AD HOC ROUTING PROTOCOLS

Certain proactive routing protocols include dynamic destination sequenced distance vector (DSDV) [24], wireless routing protocol (WRP) [25], global state routing (GSR) [26], clusterhead gateway switch routing (CGSR) [27], fisheye state routing (FSR) and hierarchical state routing (HSR) [28], landmark routing (LANMAR) [29], optimized link state routing (OLSR) [30], topology dissemination based on reverse path forwarding (TBRPF) [31], distance routing effect algorithm for mobility (DREAM) [32]. Among them, DSDV, CGSR, WRP, and DREAM belong to the family of distance-vector routing. Routing performed in DSDV and WRP is based on a flat architecture. The overhead of the route update in GSR, CGSR, LANMAR, TBRPF and FSR-HSR is reduced because of the *hierarchical architecture* used in these protocols. TBRPF reduces the route update overhead by maintaining and sharing a tree for update broadcast. In FSH-HSR and LANMAR, nodes slow down the update rate as their distances from destinations or landmark nodes increase, respectively. OLSR uses multipoint relay flooding which significantly reduce the overhead of route update by selecting only some of the neighbor nodes as relays [33]. Among them, DREAM is based on *location information*, and builds location tables by flooding position updates throughout the network. The frequency at which DREAM sends position updates is related to both mobility rate, and distance between nodes. Examples of reactive protocols are ad hoc on demand distance vector (AODV) [34], dynamic source routing (DSR) [35], temporally ordered routing algorithm (TORA) [36][37], associativity based routing (ABR) [38], signal stability routing (SSR) [39], relative distance microdiscovery ad hoc routing (RDMAR) [40], and location-aided routing (LAR) [41]. All of them are based

on a flat architecture. DSR and AODV apply expanding-ring search methods during route discovery in order to reduce the overhead of full search strategy. RDMAR, on the other hand, estimate the relative distance between the source and destination nodes, and avoids flooding by using query localization to limit the scope of the discovery. LAR is based on location information, and floods location requests instead of route requests which intend to limit the overhead of full search to the requested zone. The zone routing protocol (ZRP) [42], zone-based hierarchical link state (ZHLS) routing protocol [43], cluster based routing protocol (CBRP) [44] are three hybrid routing approaches. All of them partition the network into a set of zones. ZHLS decreases the overhead on full-search outside of a zone by maintaining the zone connectivity of the whole networks. In CBRP, the overhead of full search is reduced to a set of clusterhead. ZRP and ZHLS employ link state routing protocol within zones.

D DISCUSSION

Proactive protocols suffer from the disadvantage of additional control traffic that is needed to continually update stale route entries. Since the network topology is dynamic, when a link goes down, all paths that use that link are broken and have to be repaired. If no applications are using these paths, then the effort that went in to repair may be considered wasted. This wasted effort results in inefficient use of scarce bandwidth resources and cause further congestion at intermediate network points. Such protocols are scalable in relation to the frequency of end-to-end connections. Although proactive protocols are not scalable with respect to the total number of nodes, they can be made scalable if a hierarchical architecture is used. Finally, proactive protocols are not scalable in relation to the frequency of topology change. As a result, this strategy is more appropriate for a network with low mobility and high end-to-end connectivities. Reactive protocols may not be optimal in terms of bandwidth utilization because of flooding of the route discovery request, but they remain scalable with respect to the frequency of topology change. Such protocols are not scalable in the number of nodes, however, they can be made scalable if a hierarchical architecture is used. Further, reactive protocols are not scalable in the number of flows. Thus reactive strategies are more suitable for networks with high mobility and relatively small number of flows. Hybrid approaches try to provide a compromise on scalability issue in relation to the frequency of end-to-end connection, the total number of nodes, and the frequency of topology change.

One of the challenges in ad hoc routing is related to the underlying broadcasting technique or a derivative of it used to perform the routing functionalities. Most of the protocols both proactive and reactive employ the simplistic form of broadcasting called flooding, in which each node retransmits the unique received packet exactly once [45, 46]. This potentially generates high overhead in the network. One of the problems related to the flooding is the broadcast storm problem [47], where a node receives the same message from multiple neighboring nodes at about the same time. Observations in [47] reveal that serious redundancy, contention, and collision could exist if flooding

is done blindly. Some methods are required to damp the process of packet generation and duplication at each node. The idea is to select a subset of the set of neighboring nodes to forward a packet (i.e. selective flooding), which in turn reduces the overhead of redundant rebroadcasting. There are four algorithms for selective flooding [45]: simple flooding [46], probability based methods [47], area based methods [47], and neighbor knowledge methods [33, 48]. Simple flooding requires each node to rebroadcast all packets. Probability based methods use some basic understanding of the network topology to assign a probability to a node to rebroadcast. Area based methods assume nodes have a common transmission distance; a node will rebroadcast if the rebroadcast will reach sufficient additional coverage area. Neighbor knowledge methods maintain state of their neighborhood, using periodical hello message, which is used in the decision-making of rebroadcast. Camp et al. provide an exhaustive performance comparison between a subset of each family [45]. Another important problem in this regard is related to the *scope* of flooding, which in general is the diameter of the network. Among proactive protocols, only FSH-HSR, LANMAR, and DREAM control the scope of flooding. FSR-HSR and its descendant LANMAR limit the scope of flooding to the fisheye scope, which means to maintain accurate routing information about the immediate neighborhood of a node with progressively less detail as the distance increases, i.e. fisheye scope. DREAM also limits the scope of flooding due to the so called *distance effect*, i.e. the farther two nodes separate, the slower they seem to be moving with respect to each other. There exist several techniques to limit the scope of flooding for reactive approach including relative distance estimation used in RDMAR [40], request zone estimation used in LAR [41], expanding ring search used in DSR [49] and AODV [50], and query localization [51]. The RDMAR estimation is arrived at on the basis of the previously known value of the distance between the source and destination nodes and the time elapsed since this value was recorded. The expanding ring search strategy initially broadcasts the request to a small area and if the destination lies within that region, the query is successful. If it lies beyond the expected region, a retry is made with increased time-to-live (TTL). LAR technique uses location information (e.g. by means of global positioning system GPS) to estimate the zone in which the destination might be located. Query localization techniques use prior routing history to estimate a small region in the network with the high probability of finding the destination node.

E CURRENT IETF ROUTING PROTOCOLS

The Internet engineering task force (IETF) has formed a working group named Manet in the area of mobile ad hoc networking [1]. The purpose of this working group is to standardize IP routing protocol functionality suitable for wireless routing application within both static and dynamic topologies. The fundamental design issues are that the wireless link interfaces have some unique routing interface characteristics and that node topologies within a wireless routing region may experience increased dynamics, due to motion or other factors. The WG currently operates under a reduced scope by targeting the promotion of a number of core routing protocol specifications to EXPER-

IMENTAL RFC status namely AODV, DSR, OLSR and TBRPF. Some maturity of understanding and implementation exists with each of these protocols, yet more operational experimentation experience is seen as desirable. Overall, these protocols provide a basic set of Manet capabilities covering both reactive and proactive design spaces.

F CONTRIBUTIONS

A new architecture that separates topology management from route determination is proposed [52]. In this architecture, topology management adjusts the network topology as nodes move and environment changes; while route determination generates and selects path(s), and forwards user traffic between source and destination. This architecture optimizes network and routing performances according to two criteria: quality of connectivity and quality of service; where quality of connectivity characterizes the quality of network, and quality of service identifies the application requirements. Topology management forms a logical structure between nodes with respect to the current quality of connectivity to optimize network performance. Route determination, on the other hand, finds the most suitable path on the top of the logical structure according to desired quality of service so as to optimize routing performance.

Our research presented in this dissertation is composed of three complementary chapters tackling the problem of topology management, routing, and quality of service in the mobile ad hoc networks; respectively.

F.1 Topology Management

In the proposed topology management strategy [53, 54], each node selects a neighbor called *preferred neighbor* according to the degree of connectivity, so as to construct a preferred link. The set of preferred links in each neighborhood forms a set of preferred paths in the network, which will be used for routing. We will prove that whatever the network topology is, connecting each node to its preferred neighbor always yield to a forest. As a result, a forest of nodes with a high degree of connectivity is constructed from the network topology. This is done in a distributed fashion using only periodic message exchanges between nodes and their neighbors. The constructed forest reduces the broadcasting overhead by selecting a subset of the set of neighboring nodes for forwarding a packet. Each tree of the constructed forest forms a zone, and each zone is maintained proactively. This is desirable because it improves the delay performance within zones. In this way, the network is partitioned into set of non-overlapping dynamic zones. Zones are connected to each other via the nodes that are not in the same zone but are in the direct transmission range of each other. So, the network can be seen as a set of connected zones. A node maintains routing information only to those nodes that are within its zone, and information regarding only its neighboring zones. Hence, the reactive behavior between zones. To sum up, our algorithm combines two main notions: zone and forest. Zones are used to reduce the delay due to routing process

and to reach high scalability. Forest reduces the broadcasting overhead by selecting a subset of the set of neighboring nodes for forwarding a packet. Chapter III presents in details our topology management strategy.

As the degree of connectivity does not fully characterize the quality of connectivity, we suggest a mechanism to describe the *quality of connectivity* for extracting the *links* connecting the pair of best nodes over time from the network point of view, and use this quality as *the* criteria for preferred neighbor election [55]. This is because the performance of ad hoc routing strictly depends on the quality of each individual node. Indeed, this quality should not only reflect the available resources in a node but also the stability of such resources because of two reasons: first, mobile ad hoc networks potentially have less resources than wired networks and second, mobility may result in link failure. Therefore, more criteria are required in order to capture the quality of connectivity between nodes. We identify three metrics to represent the quality of connectivity from the network point of view: power lifetime, unallocated buffer, and stability level. By changing the criteria of preferred neighbor election from degree of connectivity to quality of connectivity, we construct a forest of nodes with the high quality links. This is desirable because the subset of the set of forwarding nodes belongs to the nodes with the high quality, which in turn improves routing performance. This constructed forest is then dynamic and adaptive because the quality of nodes change over time according to the network behavior. This issue is also addressed in chapter III.

F.2 Route Determination

We propose a hybrid ad hoc routing protocol (HARP) which combines proactive behavior within a zone and reactive behavior between zones [56]. HARP functions at two levels: zone and node depending on whether the topology management is present. By the zone level routing, we mean to conceal the zone details and look upon them as nodes. The aim here would be to establish a connection from the source node's zone to the destination node's zone. A link that connects nodes belonging to different zones is called a *bridge*. Hence, in this case, we are looking for a path of *bridge* edges that connect the zone of the source node to the zone of the destination node. Routing is performed on two phases: intra-zone and inter-zone, depending on whether the destination belongs to the same zone as the forwarding node. Intra-zone routing relies on an existing proactive mechanism inherited from the topology management algorithm and therefore, there is no route acquisition delay. Intra-zone routing is done based on the intra-zone table. Inter-zone routing, on the other hand, is reactive and applies the path discovery procedure to find the shortest path to the destination. The overhead related to flooding nature of this procedure is noticeably reduced by the forest structure since a subset of the set of neighboring nodes forwards the unique received packet. Distance estimation mechanism is also invoked to find a distance from the source to the destination, which in turn limits the scope of flooding. During the inter-zone routing procedure, each forwarding node applies the intra-zone routing as soon as it finds out that the destination node belongs to its zone. Chapter IV) presents how our routing pro-

tocol functions and takes the advantage from the presence of the topology management strategy.

F.3 Quality of Service Support

The quality of service that an application requires depends strictly on the quality of the network. Therefore, we define network metrics and application metrics as additional constraints to the conventional ones to determine paths between a source and a destination [57, 58]. Network metrics are used to determine the quality of a path to route the data traffic using the quality of individual node in that path. We define three network metrics: hop count, path buffer, and path stability. Hop count corresponds to the number of hops required for a packet to reach its destination. The two other metrics, path buffer and path stability, determine the buffer level and stability level used to describe the quality of connectivity *for a path*. The main objective of network metrics is to provide a trade-off between load balancing and resource conservation. Hop count minimizes the network resource consumption while path buffer and path stability avoid unbalanced utilization of the resources. In this sense, the network metrics are primary metrics and support the network since they are used to *generate* paths between source and destination. We define three application metrics including delay, throughput, and enhanced best-effort to meet specific application requirements. In order to be able to compute these metrics, a reasonable combination of network metrics is mapped onto the application metrics. Once the quality of paths is determined by the network metrics, the application metrics select exactly one path in order to meet the desired requirements. In this sense, the application metrics become secondary metrics and support applications. Chapter V will further elaborate our quality of service model, and its integration with our architecture.

F.4 Summary

From the technical point of view, the novelty of the contribution is that it develops and combines the notions of forest, zone, quality of connectivity, and relative distance estimation. Forest reduces the broadcasting overhead by selecting a subset of the set of neighboring nodes for forwarding a packet. Zones are used in order to reduce the delay due to routing process and to reach high scalability. The quality of connectivity extracts the links connecting the pair of best nodes for the purpose of routing. Relative distance estimation localizes route searching and limits the search area, and thus reduce the control overhead.

G COMPARISON

Table II-1 summarizes and compares the results from the qualitative analyzes of our routing protocol HARP with the current IETF routing protocols, namely AODV, DSR, OLSR, and TBRPF. The properties of each protocol is shown. Neither of the protocols support power aware routing nor security. However, the work in these areas is in progress and will probably be added to the protocols.

TABLE II-1. COMPARISON OF HARP WITH THE IETF ROUTING PROTOCOLS

Parameters vs. RP	OLSR	TBRPF	DSR	AODV	HARP
Routing Strategy	Proactive LS	Proactive LS	Reactive	Reactive	Hybrid
Net. Architecture	Flat	Hier. Tree	Flat	Flat	Hier. Zone
Beaconing	Yes	Yes	No	Yes	Yes
Periodic Update	Yes	Yes	No	No	No
Ctrl Flooding OH	Yes	Yes	No	No	Yes
Ctrl Flooding Scope	No	No	Yes	Yes	Yes
QoS Support	No	No	No	No	Yes
Multicast Support	Yes	Yes	No	Yes	Yes [59]
Power Management	No	No	No	No	No
Security support	No	No	No	No	No

CHAPTER III

Topology Management in Mobile Ad Hoc Networks

Contents

A Motivation 22
B Basic Idea 23
C Preliminary Definitions 24
D DDR – Distributed Dynamic Routing Algorithm 26
D.1 Preferred Neighbor Election 27
D.2 Forest Construction 29
D.3 Intra-zone Clustering 30
D.4 Inter-zone Clustering 32
D.5 Zone Naming 33
D.6 Zone Partitioning 35
D.7 Zone Maintenance 35
D.8 Summary 36
E Correctness 36
F Pseudo Code and Flow Chart 37
G Mathematical Analysis 38
H New Criteria for Preferred Neighbor Election 42
H.1 Quality of Connectivity at the Network Layer 42
H.2 Forest of High Quality Nodes 44
I Behavioral Results on Topology Management Strategy 44
J Protocol Model 48

K Simulation Model 49
L Performance Results 50
L.1 Packet Delivery Fraction 50
L.2 Average end-to-end delay of data packets 50
L.3 Routing overhead 52
M Architecture 52
N Conclusion 57

Abstract— *An architecture that separates topology management from route determination is proposed. In this architecture, topology management adjusts the network topology as nodes move and environment changes; While route determination generates and selects path(s), and forwards user traffic between source and destination. This architecture optimizes network and routing performances according to two criteria: quality of connectivity and quality of service; where quality of connectivity characterizes network quality, and quality of service identifies application requirements. Topology management forms a logical structure between nodes with respect to the current quality of connectivity. Route determination, on the other hand, finds the most suitable path according to the desired quality of service. We first characterize the behavior of the proposed topology management strategy under various network density. A detailed simulation model is used to study the effect of our topology management on the performance of routing. We demonstrate that this architecture provides an optimal trade-off between routing overhead and delay experienced by routing protocols, and it does improve routing performance.*

Keywords— *Mobile ad hoc networks, architecture, routing, topology management, forest, zone, quality of connectivity, simulation, performance evaluation.*

A MOTIVATION

MOBILE ad hoc networking *Manet* is a challenging task due to frequent changes in network topology as well as the wireless nature of the network interface. The frequent changes in network topology imply a limited lifetime for the topology information as a function of mobility rate. Such information has to be periodically updated in order to remain valid. Thus, the more frequent the information is updated, the better the node mobility can be managed. On the other hand, the wireless nature of the interface implies limited bandwidth capacity in the network. Furthermore, wireless links experience high collision probability due to their broadcast nature, as well as high bit error rate. Therefore, the mobile and the wireless environments exhibit opposite requirements. Indeed, the mobility requires more information to be send to keep track of the network changes, while the wireless medium has low capacity and hence can not be used for additional control traffic that is needed to continually update stale information.

Routing protocols for mobile ad hoc networks have the twin design goals of low route delay and low control overhead. The former is important because it contributes to the

delay encountered by data packets. The control overhead is very critical as far as efficient utilization of network resources is concerned. In the ad hoc networking scenario, it is difficult to minimize both the delay and the overhead simultaneously and therefore, some degree of trade-off is always required. Proactive routing has low delay but significantly large control signaling is needed to maintain exhaustive routing tables [30]. Reactive protocols try to reduce control overhead by discovering routes on-demand but the discovery procedure increases the delay [35, 34]. Our primary *goal* is to provide a trade-off between overhead and delay of a routing protocol while maximizing the network resource utilization. In order to achieve a trade-off between routing overhead and delay while maximizing network resource utilization, we apply an architecture that benefits from the separation between topology management from route determination. This is desirable because topology management dynamically structures the network topology as a function of node mobility and network traffic load; and route determination finds the most suitable path on this structure according to application requirements. In this chapter, we only address the topology management strategy and its effect on routing. Therefore, we aim at achieving that trade-off rather than maximizing the network resource utilization.

Mobile ad hoc networks experience link failure more often because of node mobility and multihop nature of routing as well as lack of resources in the network. Hence, a routing protocol should consider the reasons for link failure to improve its performance. Link failure stems from node mobility and lack of resources. Therefore it is essential to capture the aforesaid characteristics to identify the *quality of connectivity* between nodes so as to decrease the probability of such failures. Thus, our second *goal* is to characterize the quality of connectivity and use this to improve routing performance.

The rest of this chapter is organized as follows. In the following sections, we highlight the basic idea of our approach. Then, we outline the necessary preliminary definitions to describe the algorithm. Afterward, we present in detail the seven phases of our topology management strategy and its correctness. The pseudo code of the algorithm shows how the algorithm works. We introduce the concept of quality of connectivity over time so as to adapt our algorithm to the quality of network. We characterize the behavior of the algorithm under various network densities. A detailed simulation model is used to study the effect of our topology management on routing. Finally, we draw concluding remarks and highlight some future work.

B BASIC IDEA

In our algorithm [53], each node selects a neighbor called *preferred neighbor* according to the highest degree of connectivity (a unique criteria) so as to construct a preferred link. The set of preferred links in each neighborhood generates a set of preferred paths in the networks, which will be used for routing. We will prove that whatever the network topology is, connecting each node to its preferred neighbor always yield a forest (c.f. section E). As a result, a forest of nodes with the high degree of connectivity is

constructed from the network topology. This is done in a distributed fashion using *only* periodic message exchanges between nodes and their neighbors known as *beacon*. The constructed forest reduces the broadcasting overhead by selecting a subset of the set of neighboring nodes for forwarding a packet. Each tree of the constructed forest forms a zone, and each zone is maintained proactively. This is desirable because it improves the delay performance within zones. In this way, the network is partitioned into set of non-overlapping dynamic zones. Zones are connected to each other via the nodes that are not in the same zone but are in the direct transmission range of each other. So, the network can be seen as a set of connected zones. A node maintains routing information only to those nodes that are within its zone, and information regarding only its neighboring zones. This algorithm is denoted by DDR which stands for distributed dynamic routing.

The novelty of the algorithm is the combination of two classical notions: *forest* and *zone*. Forest was previously used in DST - distributed spanning trees for routing in mobile ad hoc networks [60]. The concept of zone was also used in zone routing protocol (ZRP) [42] [61], and zone-based hierarchical link state (ZHLS) routing protocol [43]. Although DDR benefits from classical concepts like zone and forest, unlike previous solutions it achieves several goals at the same time. Firstly, it provides different mechanisms to drastically reduce routing complexity and improve delay performance. Secondly, it dynamically adapts to the network conditions, i.e. network load, mobility, and resources. Finally, it is infrastructure-less in a strong sense: it does not even require a physical location information.

C PRELIMINARY DEFINITIONS

A Manet topology is represented by an arbitrary undirected graph $G = (V, E)$, where V is the set of mobile nodes (MN), and E is the set of edges. An edge exists if and only if the distance between two MNs is less or equal than a fixed radius r. This r represents the radio transmission range which depends on the transmission power. Accordingly, the neighborhood of a node x is defined by the set of nodes that are inside a circle with center at x and radius r, and it is denoted by $N_r(x) = N_x = \{n_j | d(x, n_j) \leq r, j \in \mathrm{N}, j \leq |V|\}$, where x is an arbitrary node in graph G. We assume that nodes are moving in a two-dimensional plane. The degree of node x in G is the number of edges which are connected to x, and it is equal to $deg(x) = |N_r(x)|$. The graph $G = (V, E)$ is called a tree T if and only if G is connected and contains no cycles. A node becomes a leaf node in the tree T if its degree is equal to 1. A forest F is a graph whose connected components are trees. We assume that each mobile node x generates periodically a message, known as the *beacon* B, to the neighboring nodes that are within its direct radio transmission range. The beacon is used to construct a forest in a distributed way. There are five fields in a beacon: Zone ID number (ZID), Node ID number (NID), degree of NID (NID_DEG), is this neighbor my preferred neighbor (MY_PN) and the ID of the preferred neighbor(s) (PN_ID). The beacon B of node x is shown in Fig. III-1, and it is denoted by B_x.

Fig. III-1. Fields of a beacon

The ZID is the ID of a tree. Each node initiates its ZID to its NID. The ZID is determined depending on the ID numbers of some selected nodes in a tree. The way in which these nodes are selected, will be explained in section D.5. The ZID is also used to distinguish the nodes that are not in the same tree but are in the direct transmission range of each other. These nodes are called *gateway* nodes, and the edge that connects two gateway nodes is called *bridge*. A node is in *non-router* mode, if it is a non-gateway leaf node. Each MN is identified by a unique ID number called NID. The NID_DEG is the degree of the NID in graph G. Each node calculates its degree by counting the number of different received beacons during a period, that is $deg(x) = |B_{N_r(x)}|$. The MY_PN flag distinguishes two different modes: PN election mode and PN forward mode. The PN election mode indicates whether the preferred neighbor is determined by x; in this case this flag is set to 1. The PN forward mode indicates that node x notifies the nodes belonging to its tree about *new* or *removed* member(s); in this case the flag is set to 0. The preferred neighbor of x is the node with the maximum connectivity in the neighborhood. Each node selects only one preferred neighbor (PN) and can be chosen as the preferred neighbor of many nodes. The way in which a PN is chosen will be explained in section D.1.

Each node in the network maintains three tables: neighboring table, intra-zone table, and inter-zone table. Basically, neighboring table is the table through which node x detects changes to its neighborhood. It has three fields: node ID number (NID), and node degree (DEG), and last update time (LUT). The NID represents the ID number of the neighbor, and the DEG represents its degree of connectivity in the neighborhood. Such information is considered valid for a limited period of time due to node mobility, and must be refreshed periodically to remain valid. The LUT represents the last update time of the NID, and it is used to purge the expired neighbor from the table. Note that, we assume each node periodically sends a beacon to its neighborhood indicating its presence and its degree of connectivity. Table III-1 represents the neighboring table of node x: NT_x.

TABLE III-1. NEIGHBORING TABLE

NID	DEG	LUT

Intra-zone table keeps the information regarding the nodes belonging to the same tree. It contains three fields: node ID number (NID), learned preferred neighbors (Learned_PN), and last update time (LUT). The NID represents the ID number of a neighbor that holds a direct tree edge with node x. So, a node becomes a *leaf node* if it has only one entry in this table. The Learned_PN represents the nodes that are learned

by their corresponding NID and are reachable through them. The LUT is used to purge the expired entry corresponding to NID from the table. The intra-zone table of node x is shown in Table III-2, and it is denoted by $Intra\_ZT_x$. The $Intra\_ZT_x$ gives the current view of node x about its tree, and it is updated upon receiving beacons.

TABLE III-2. INTRA-ZONE TABLE

NID	Learned_PN	LUT

On the contrary, inter-zone table keeps the information concerning neighboring zones of the zone to which node x belongs. Each entry in this table contains the ID number of gateway nodes (GNID), the zone ID of such gateway nodes, i.e. neighboring ZID (NZID), and neighboring zones stability regarding node x (Zone_Stability), and the last update time (LUT) of this entry which is used as in the intra-zone table. A node is a *gateway node* if its inter-zone table contains at least one node. The Zone_Stability field represents the stability of the neighboring zones regarding node x, and it is update upon receiving a beacon. The inter-zone table of node x is shown in Table III-3, and it is denoted by $Inter\_ZT_x$.

TABLE III-3. INTER-ZONE TABLE

GNID	NZID	Zone_Stability	LUT

D DDR – DISTRIBUTED DYNAMIC ROUTING ALGORITHM

The DDR - algorithm consists of seven cyclic time-ordered phases: (1) preferred neighbor election, (2) forest construction, (3) intra-zone clustering, (4) inter-zone clustering, (5) zone naming, (6) zone partitioning, and (7) zone maintenance which are executed based on the information provided by beacons. A beacon is a periodic message exchanged *only* between a node and its neighboring nodes. The content of a beacon is primitive at the beginning, and it will be enriched during subsequent phases of the algorithm. During the beaconing process, each node gathers the information describing its neighborhood in its neighboring table. Then, each node in the network topology chooses a neighbor whose degree of connectivity is equal to the maximum neighborhood degree. This neighbor is called preferred neighbor, and it is used to construct a preferred link between nodes for the purpose of routing. Then, a forest is constructed by connecting each node to its preferred neighbor and vice versa. We prove in section E that whatever is the network topology, connecting each node to its preferred neighbor always yields a forest, i.e. we have no cycle. Afterward, the intra-zone clustering algorithm develops an appropriate structure from each tree, and builds the intra-zone

routing table. The inter-zone clustering algorithm provides the inter-zones connections which are kept in the inter-zone routing table. Since the forest is abstracted into a set of non-overlapping zones, hence the network is partitioned. Finally, the zone maintenance phase updates the forest and the zones according to the preferred neighbor election phase algorithm. It has to be mentioned that the algorithm only uses beacons to perform every seven phases of the algorithm. Therefore, it avoids global broadcasting throughout the network. Fig. III-2 shows the different phases of the algorithm.

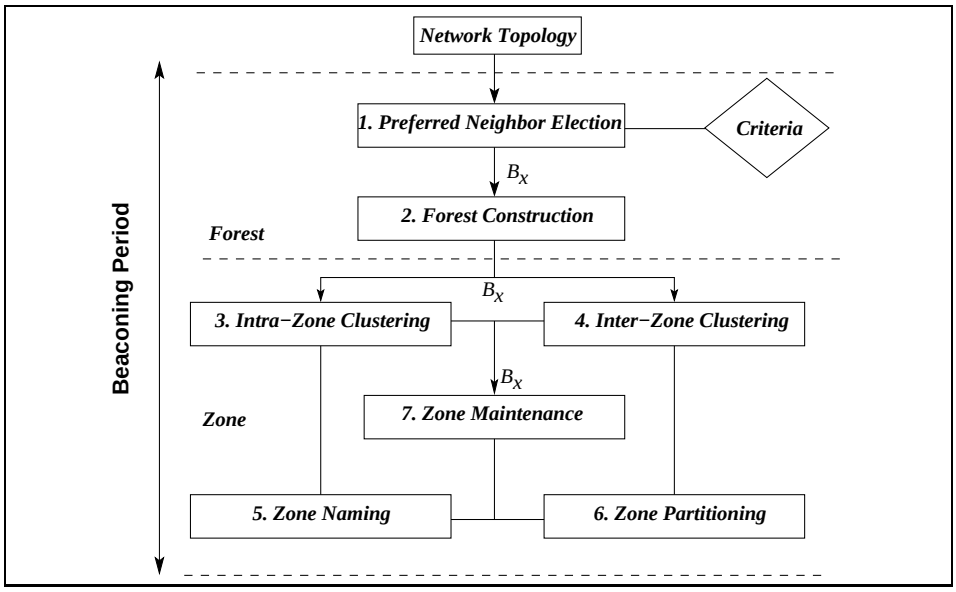


Fig. III-2. Diagram illustrating the different phases of the algorithm DDR

D.1 Preferred Neighbor Election

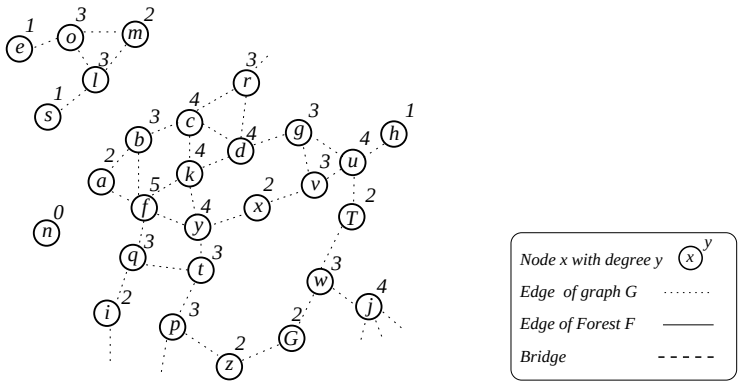
Let x and y be a node of the graph $G = (V, E)$. We assume that each node transmits a periodical beacon to its neighborhood indicating its degree and its ID; and upon receiving beacons, each node stores their information in the neighboring table. Fig. III-3 shows an arbitrary graph, where each circle represents a node with its ID number and its degree, and each dotted lines represents an edge (or a link) between two nodes in the transmission range of each other. Table III-4 gives a view of neighboring table of node k in this figure. Based on the neighboring information, node x determines its preferred neighbor (PN). For this purpose, node x computes a set of nodes whose degrees are equal to maximum neighborhood degree. This set is denoted by $PN_x = \{y | y \in N_x \wedge deg(y) = \max(deg(N_x))\}$. We distinguish three cases.

- ***No PN***— if the set is empty, then node x has no PN which means it has no neighbors. In Fig. III-3, node n has no neighbor and consequently no PN;

- ***Single PN***— if PN_x has only one member, then this member is the elected PN. For example, in Fig. III-3, node k has four neighbors: f, c, d, y, but the set PN_k only includes f;

- ***Multiple PN***— the set PN_x can have more than one member which is the case for node f, since $PN_f = \{k, y\}$. This means that there are more than one neighbor with the maximum neighborhood degree. In this case, we assume that node x elects a node with the greatest ID number. So, node f elects node y since its ID number is greater than node k (regarding to the alphabetical order).

Therefore, each node selects a neighbor that has maximum connectivity in the neighborhood. For nodes that evaluate two identical degree of connectivity, we break ties by setting the convention that nodes with higher IDs are preferred. We say that node y is the *preferred neighbor* of node x, if y is in the neighborhood of x and has the maximum connectivity among its neighbors. In this manner, each node in the network can only choose one preferred neighbor, and can be chosen as a preferred neighbor of many nodes. Section H.1 introduces alternative criteria for preferred neighbor election algorithm, which incorporate link quality.



(a) An arbitrary graph G (b) Legend

Fig. III-3. Each node in the graph is characterized by its degree and a letter which represents its ID number, and we assume that each node knows the ID numbers and the degrees of its neighboring nodes

TABLE III-4. NEIGHBORING TABLE OF NODE k

NID	DEG	LUT
f	5	t_f
c	4	t_c
d	4	t_c
y	4	t_y

D.2 Forest Construction

The forest is constructed by connecting each node to its PN, as shown in Fig. III-4. In section E, we will prove that, whatever is the network topology, this approach always yields a forest (i.e. no cycle). This is because of the way in which a node is elected follows a *monotonic increasing function* depending on the degree and on the ID number. In the Fig. III-4, the solid lines represent tree edges, and the dashed lines represent the edges that connect nodes belonging to different trees (see Fig. III-3(b)).

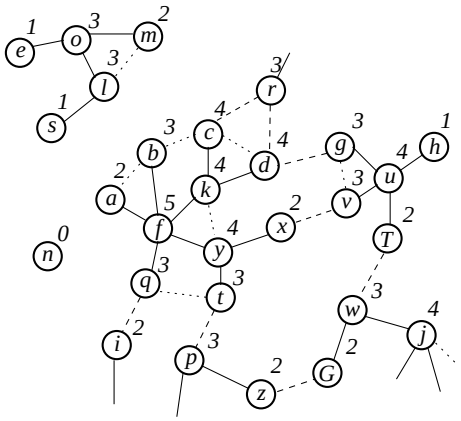


Fig. III-4. Constructed forest

In order to construct preferred links and consequently the forest, each node generates a table called *intra-zone table*. This table maintains the tree members, and is updated periodically upon receiving beacons. Indeed, as soon as node x determines its PN y, it must notify its neighboring nodes, especially y, of its decision. Therefore, node x sets its beacon to $B_x = (ZID, x, deg(x), 1, y)$, and updates its intra-zone table regarding y. This beacon indicates that node x from zone *ZID* with the degree $deg(x)$ is electing $< 1 >$ node y as its PN, and we call it a beacon in election mode. It has to be mentioned that the beacons in election mode are transmitted *periodically*. Upon receiving x's beacon, node y verifies whether it has been chosen as the PN of x. If

so, it also updates its intra-zone table regarding x. This means that a tree edge is built between node x and its preferred neighbor y, and thus node x and y belong to the same tree. Therefore, this edge becomes a preferred link, and the set of preferred links in each neighborhood generates the set of preferred paths in the networks.

For example in Fig. III-4, node k elects node f as its PN due to f's highest degree in k's neighborhood. Then it generates the beacon $B_k = (ZID, k, 4, 1, f)$ in election mode, and inserts node f into the NID field of its intra-zone table. Upon receiving k's beacon, node f inserts node k into its intra-zone table $Intra\_ZT_f$. Therefore, the link between node k and f becomes a preferred link. Node c and d will elect node k as their PN because of k's highest ID in c and d's neighborhood. The node $y, b, q,$ and a elect node f as their PN. Note that, node f chooses node y as it PN but it has been chosen as the PN of k, b, a, q and y. Table III-5 shows intra-zone tables of node k and f. Note that, at this stage there is no learned PN because nodes are in the early stage of the algorithm, i.e. PN election phases.

TABLE III-5. INTRA-ZONE TABLE OF NODES k AND f REGARDING FIG. III-4

NID	Learned_PN	LUT
f	-	t_f
c, d	-	t_c, t_d

(a) $Intra\_ZT_k$

NID	Learned_PN	LUT
y	-	t_y
k	-	t_k
b, a, q	-	t_b, t_a, t_q

(b) $Intra\_ZT_f$

Indeed, this forest can be tuned via the *criteria* of the preferred neighbor election in order to achieve the desired requirements. For instance in the algorithm, we may also benefit from minimum neighborhood connectivity or maximum neighborhood stability in conjunction with maximum neighborhood connectivity, as a second criteria for preferred neighbor election, which in turn provides other alternative routes to the destination node. These alternative routes can potentially balance the load in the network. Hence, by changing or by adding a new criteria to the preferred neighbor election algorithm (c.f. Fig. III-2), one can construct an alternative forest. In section H, we introduce a novel combined criteria to adapt the forest to the quality of network. Fig. III-5 shows the forest constructed by the minimum neighborhood connectivity.

D.3 Intra-zone Clustering

In this phase, nodes attempt to expand their own view about the tree they belong to by completing their intra-zone table. For this purpose, each node locally advertises the new PN learned during the forest construction phase by means of a new type of beacon called beacon in *forward* mode. Upon receiving this beacon, each tree member

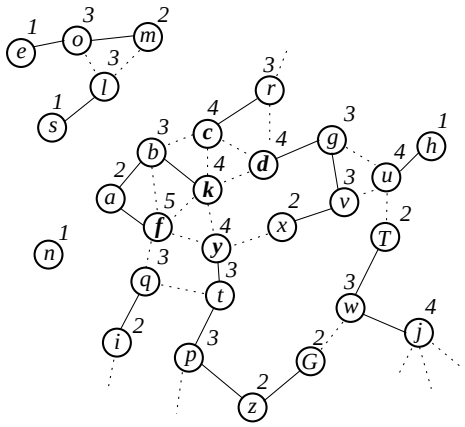


Fig. III-5. Constructed forest with minimum neighborhood degree

determines the *new* PNs learned from that neighbor and re-advertises to their neighbors if they are not leave nodes. Recall from the previous sections that, node y is chosen to be the PN of x, and x has sent a beacon to inform its neighborhood of its elected PN. Among the neighboring nodes of x, the PN y forwards x's decision to nodes that hold a tree edge with y [1] by setting its beacon to $B_y = (ZID, y, deg(y), 0, x)$. If node y is chosen as the PN of many nodes through a period, then y forwards their decisions encapsulated in PN field in the beacon, that is $B_y = (ZID, y, deg(y), 0, x : x' : x'' : ...)$. This beacon indicates that node y from zone *ZID* with the degree $deg(y)$ is forwarding $< 0 >$ the new learned PNs or tree members $< x : x' : x'' : ... >$ to the rest of tree, and we call it a beacon in forward mode. A beacon in *forward* mode reflects that the information carried by that beacon is a propagation of some information that the sender of the beacon has just received. This beacon is a *non-periodical* beacon. Other neighboring nodes of x add y to the Learned_PN field corresponding to x in their intra-zone tables if they belong the same tree as node x (i.e node x already exists in their tables). In this way, we say that y is learned to be the PN of x. Note that node x is also learned by the neighboring nodes of y. Node x forms a beacon in the PN forward mode if the set of PN learned by x is non-empty. This set is denoted by $Learned\_PN_x$. Node x forwards the $Learned\_PN_x$, if it is not a leaf node. Hence, the intra-zone table is only updated if the information is originated by the PN members and not by the Learned_PN members or by a normal neighbor.

For example in Fig. III-4, node k elects node f as its PN. So, node f can be learned by nodes c, d. Upon receiving k's beacon, node f updates the information regarding node k in $Intra\_ZT_f$. Since nodes b, a, q, y, k choose node f as their PN, node f must forward their decisions encapsulated in PN-field by setting its beacon to $B_f = (f, 5, 0, b : a : q : y : k)$. Therefore, nodes b, a, q, y are learned by node k, i.e. $Learned\_PN_k = b, a, q, y$. So, the set of $Learned\_PN_k$ will be learned by the nodes

[1] These nodes reside in the first column of intra-zone table of node y, i.e. $Intra\_ZT_y.NID$.

c & d as well. But, nodes c, d do not forward their learned PN, since they are leaf nodes. Note that, nodes a and b are in the non-router mode, because they are non-gateway leave nodes. Table III-6 illustrates intra-zone tables of nodes f and k, when their tree is established.

TABLE III-6. INTRA-ZONE TABLE OF NODES k AND f REGARDING FIG. III-4

NID	$learned\_PN$	LUT
f	a, b, q, y, t, x	t_f
c, d	-	t_c, t_d

(a) $Intra\_ZT_k$

NID	$learned\_PN$	LUT
y	x, t	t_y
k	c, d	t_k
b, a, q	-	t_b, t_a, t_q

(b) $Intra\_ZT_f$

As shown in Table IV-2, the view of node x about its tree consists of two levels: NID, $Learned\_PN$. The NID level contains the nodes holding tree-edges with node x, i.e. node x can reach them directly. The second level $Learned\_PN$ contains the nodes that are learned by the NID level. In fact, node x can reach them via their corresponding NID. Therefore, node x only knows the next hop for its second level nodes. Actually, each entry in $Intra\_ZT_x$ can be seen as a branch of x, that is $NID \rightarrow Learned\_PN$. Thus, each node obtains a partial view of its tree in the sense that it does not know the detailed structure of its tree. For example in Fig. III-4, consider the scenario where node k wants to communicate to one of the nodes belonging to its tree. According to its intra-zone table (see Table IV-2), node k can reach the nodes a, b, q, y, t, x through node f, while other nodes f, c, d are directly reachable. So, regarding the $Intra\_ZT_k$, the next hop to reach the nodes a, b, q, y, t, x is the node f and not c, d. Fig. III-6 shows the view of node k on its tree.

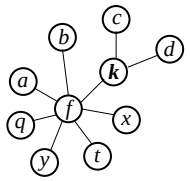


Fig. III-6. View of node k on its tree

D.4 Inter-zone Clustering

Node x encounters two cases during the construction of its tree. Either it can succeed to add some nodes to its tree and updates its intra-zone table; or node x puts the remaining nodes in its inter-zone table. These nodes are considered as gateway nodes and they will be moved from the inter-zone table to intra-zone table whenever they can join the

tree. For example in Fig. III-4, node c belongs to the inter-zone table of node d until node d is informed about the existence of c in the tree. After that, node d moves node c from its inter-zone table to its intra-zone table. Table III-7 shows the inter-zone table of node d, when the zone is fully established (see Fig. III-7). The Zone_Stability field represents the stability of the neighboring zone regarding node x, and it is incremented if the currently received ZID of that gateway node is similar to the old one. Indeed, the stability of a zone depends directly on the ZID number. Section D.5 provides a detailed explanation of how a node determines its ZID and when it increments the Zone_Stability of its neighboring zone.

TABLE III-7. INTER-ZONE TABLE OF NODE d

GNID	NZID	Zone_Stability
r	z_4	++
g	z_5	++

D.5 Zone Naming

Zone naming phase enables each node to determine a unique ID for the zone without any communications. In fact, each zone member computes its own zone ID independently, and each of them arrives at nearly the same ZID. The zone ID should depend on some characteristics of nodes belonging to the zone such as node ID number, degree and/or stability during their life time in the zone. The choice of ID number is much simpler and does not require maintaining other information in intra-zone table. However, node degree and node stability are more pertinent than ID number regarding zone characteristics. Moreover, the zone name should not vary so often as nodes move. For this purpose, node x selects a subset s of the set of nodes in its intra-zone table for assigning a name or more precisely an ID number to the zone. Therefore, node x selects q nodes with the highest ID numbers in the zone, and sorts them in ascending order. Then, node x computes a hash function on the ID number of each selected node separately. A hash function projects a value from a set s_1 with many (or even an infinite number of) members to a value from a set s_2 with a fixed number of (or fewer) members. Afterward, it concatenates all the hashed ID numbers together. The outcome of this concatenation gives the ZID, and can be formally represented as:

$$ZID = h(x_q)|h(x_{q-1})|...|h(x_1),\ where\ \ x_q > x_{q-1}, q \in N,\ \&\ q = \lfloor \frac{n}{d} \rfloor \quad \text{(III-1)}$$

where $|$ denotes concatenation, n is the number of bits used to represent the ID numbers and d is the compression degree of the hash function. The compression degree of a hash function is the ratio of $|s_1|$ to $|s_2|$, that is $d = \lfloor \frac{|s_1|}{|s_2|} \rfloor$. If the cardinality of a zone (the

number of nodes in that zone) is less than q then the selected nodes use a well-known mask instead, or simply reuse the selected ID numbers respectively. Clearly, each zone will change its ID number over time. If we denote $ZID(t_1)$ the ZID at time t_1 and the $ZID(t_0)$ the ZID at time t_0 where $t_0 < t_1$, then a node uses a similarity function based on Hamming Distance to update the value of $Zone\_Stability$. This is defined as the number of hashed IDs which differ between two zone IDs. If the distance between two ZIDs is far from a predefined critical distance d_{critic}, the node sets the value of $Zone\_Stability$ to 1, otherwise, this value is incremented. This function is called $Distance$, and determines the distance between two zone IDs as follows

$$Zone\_Stability = \begin{cases} ++ & If \quad Distance(ZID(t_1), ZID(t_0)) \leq d_{critic} \\ 1 & Otherwise \end{cases} \quad \text{(III-2)}$$

$$Where \quad Distance(ZID(t_1), ZID(t_0)) = \sum_{i=0}^{i<q} \mid ZID_i(t_1) - ZID_i(t_0) \mid \quad \text{(III-3)}$$

In both cases a node updates the ZID. So, we can conclude that the zone stability is strictly related to the ZID number. Indeed, ZID determination is based on some randomly chosen NIDs in a tree. It therefore identifies the zone and it can simply reflect the zone stability. Fig. IV-1 shows the situation where each node assigns a name to its tree.

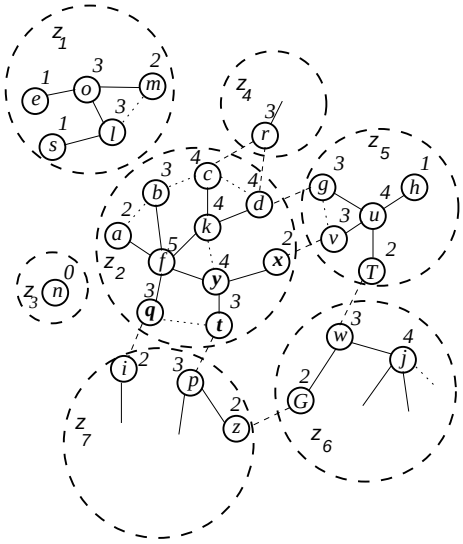


Fig. III-7. Zone naming

For example in Fig. III-7, if we assume that the $n = 12$ and $d = 3$ then $q = 4$, so node k selects the following four nodes y, x, t, q. Notice that, node x sorts these nodes in

ascending order (here they are sorted in alphabetical order). Node k determines its ZID by computing $h(y)|h(x)|h(t)|h(q)$. Note that in this way, we obtain the same number of bits in ZID as in NID.

D.6 Zone Partitioning

The forest associated to the network contains a set of trees, $T_1, T_2, ..., T_k$, where each tree form a zone and becomes an independent entity. This is because each zone is maintained proactively and is assigned with a name. Therefore, the network is partitioned into a set of non-overlapping dynamic zones, $z_1, z_2, ..., z_n$, which are connected via gateway nodes. Fig. III-8 shows the reduced graph obtained from the network topology under the algorithm.

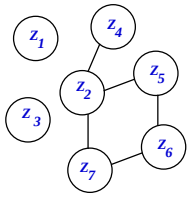


Fig. III-8. Zone partitioning (reduced graph)

Indeed, we conceal the zones' details and look upon them as nodes. The aim here would be to establish a connection from the source node's zone to the destination node's zone. A link that connects nodes belonging to different zones is called a *bridge*. Therefore, in this case, we are looking a path of *bridge* edges that connect the zone of the source node to the zone of the destination node.

D.7 Zone Maintenance

The topology information provided by the algorithm is strictly related to the preferred neighbor election algorithm. Such information may change as the preferred neighbor changes. There exist two cases where the topology changes: (i) the current PNs are expired, and (ii) a new PN is determined.

To handle the former case, node x periodically determines the expired PNs; and once a PN expires, say y, the entire entry expires with the PN, i.e. $y \rightarrow Learned\_PN$, and hence they are removed from the table. If node x succeeds to remove one or several PNs and yet it is not a leaf node, then it generates a *remove* beacon to notify the remained tree members about the removed members. A remove beacon encapsulates the removed entries in the PN field of a beacon. For this purpose, node x sets its beacon to $B_x = (-1, x, deg(x), 0, y \rightarrow Learned\_PN)$, where $ZID = 1$ means that

each node has to remove the expired PNs and recalculate the ZID. Upon receiving x's beacon, each neighbor updates its intra-zone table accordingly; and generate a new remove beacon if they are not leaf nodes.

In the latter case, once node x determines its *new* PN y', it verifies as to whether it has been chosen as the PN of y. If so, i.e. $PN_y == x$; no remove beacon is sent and the PN y remains in the intra zone table of node x since node x is the preferred neighbor of node y. Then, node x sends a beacon in election mode as it is explained in section D.2, and the new PN of node x will be learned by the zone and specially by the node y. Otherwise, if node x has not been chosen as the PN of y, then x removes the entire branch corresponding to its old PN y, i.e. $y \rightarrow Learned\_PN$, and notify the remained members of its tree about the removed members as in the former case. Afterward, node x sends a PN election beacon for its new PN. We recall that each node periodically performs the preferred neighbor election.

For example in Fig. III-7, if the link between nodes k and f is broken, then node k removes node f and its learned PNs $f \rightarrow a : b : q : y : t : x$ from its intra-zone table (see Table IV-2). Then, node k forwards the beacon $B_k = (-1, k, 3, 0, f \rightarrow a : b : q : y : t : x)$ to the remained nodes c and d in the NID field of its intra-zone table. The nodes c and d do not forward the beacon, since they are leaf nodes. Also, node f cuts the branch corresponding to $k \rightarrow c : d$, and notifies the b, a, q, y about cut branch, and so on.

D.8 Summary

The algorithm maintains information regarding the network topology by means of three tables: neighboring table, intra-zone table and inter-zone table. The neighboring table contains the set of nodes with which there exist a direct link over which data may be transmitted (in either or both directions). The intra-zone table of a particular node carries information regarding all the nodes that are belonging to the same zone as the node. So, all nodes in the same zone have readily established paths to each other. The inter-zone table of a particular nodes contains information regarding the neighboring zones. To sum up, our algorithm combines two main notions: zone and forest. Zones are used in order to reduce the delay due to routing process and to reach high scalability. Forest reduces the broadcasting overhead by selecting a subset of the set of neighboring nodes for forwarding a packet.

E CORRECTNESS

Theorem 1: For any graph G (i.e. network topology), let G' be the subgraph obtained by connecting each node to its PN. Then G' is a forest.

Proof: Let $G = (V, E)$ be the original graph and let $G' = (V, E')$ be the graph obtained by executing the algorithm in Fig. III-4 for each node $x \in V$. We first recall that the main idea of the algorithm is to select, for each node x in G, a neighbor that has maximum degree. Moreover, if more than one neighbor has maximum degree (i.e. $|PN_x| > 1$) then we select the one with maximum ID number. In order to prove that G' does not contain any cycle, we consider the function:

$$label(x) = \max_{w \in PN_x} \{deg(w)|NID\}$$

where | represents concatenation. We prove that G' cannot contain any cycle $C = x_1, \ldots, x_k, x_1$. Suppose the contrary, and let x_i be the vertex of C with the smallest value of $label(\cdot)$. Note that such a vertex is unique.

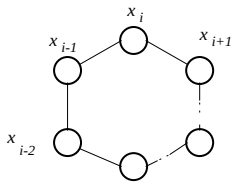


Fig. III-9. The proof of Theorem 1

Let us consider the two vertices of x_{i-1} and x_{i+1} adjacent to x_i in C (see Fig. III-9). Without loss of generality, assume that the algorithm chooses an adjacent vertex x_{i+1} (if neither x_{i-1} nor x_{i+1} are chosen, then C is not a cycle). Consider now the execution of the algorithm on x_{i-1}. We show that such a node will not choose x_i, thus implying that C is not a cycle. Indeed, by the choice of x_i, the vertex x_{i-2} adjacent to x_{i-1} satisfies one of the following two conditions:

1. $deg(x_{i-2}) > deg(x_i)$. In this case $x_i \notin PN_{x_{i-1}}$
2. $deg(x_{i-2}) = deg(x_i)$ and $NID(x_{i-2}) > NID(x_i)$. If $x_i \in PN_{x_{i-1}}$, then also $x_{i-2} \in PN_{x_{i-1}}$ (otherwise x_i is not selected as neighbor of x_{i-1}). Since $label(x_i) < label(x_{i-2})$, $NID(x_i) < NID(x_{i-2})$ holds. So, vertex x_{i-1} will select x_{i-2}.

This proves the theorem. ■

F PSEUDO CODE AND FLOW CHART

The outline of the DDR-algorithm used to construct the forest and zones is formally stated in Pseudocode 1). This algorithm is executed at each node in the network. A

beacon in *election* mode is an announcement being made by the sender about its election of a (new) preferred neighbor. This beacon is a periodical beacon. The complete flowchart for processing a beacon arriving in *PN Elect* mode is given in Fig. III-10. The complete flowchart for processing a beacon arriving in *PN Forward* mode is given in Fig. III-11.

G MATHEMATICAL ANALYSIS

It is important to compare the communication complexity of the algorithm for topology creation [62]. The communication complexity describes the average number of messages required to perform a protocol operation. Note that this comparison does not include the complexity of path determination. This issue will be covered in the next chapter. We have selected three related protocols: LSR, OLSR and ZHLS and we have compared their complexity with DDR. Consider a network with N nodes. Let τ be the rate of topology control generation, and p be the period of beaconing or hello transmission.

In LSR, each node will generate one link state packet (LSP), and every other node has to forward it once. Therefore, the total amount of communication overhead generated by LSR S_{LSR} is

$$S_{LSR} = \tau N^2 \; messages \tag{III-4}$$

The OLSR also requires the hello messages to be exchanged between a node and its neighboring nodes, and thus this amount is

$$S_{OLSR} = pN + \tau R_N N \; messages, \;\; R_N \ll N \tag{III-5}$$

, where R_N is the average number of retransmission in an MPR flooding [33].

However in ZHLS, the network is partitioned into M zones, and each zone will have N/M nodes. The amount of communication overhead for a node becomes $(N/M)^2$ messages per zone, or $M(N/M)^2 = N^2/M$ in the network. As each zone generates one zone message and every node has to forward all zone messages once, the amount of communication overhead for a zone becomes NM. So, the total amount of communication overhead generated by ZHLS for creating topology is

$$S_{ZHLS} = \tau(N^2/M + NM) \; messages \tag{III-6}$$

Similar to ZHLS, in DDR the network is partitioned into M zones, and each zone will have N/M nodes. DDR requires periodical beacons to build and maintain the forest. The amount of communication overhead to build the forest is N since upon sending a PN election beacon a forest edge will be constructed. To construct the zones, each

Pseudocode 1 execution of DDR at node x

Let $G = (V, E)$ be a graph.
Let x be any node of G.
Let y be the old preferred neighbor of x.
Let y' be the new preferred neighbor of x.
Let B_x be the beacon of node x.
Let N_x represents the set of neighboring nodes of x.
Let B_{N_x} represents a set of beacons received by x from N_x.
Let PN_x be the set of nodes with $\max(deg(N_x))$.
Let $PN(y)$ be the preferred neighbor of y'.
Let NT_x be the neighboring table of x.
Let $ZoneTables_x$ represents the intra- and inter-zone table of node x.
Let ZID be the zone ID number of node x, initialized to x.

$NT_x\_CheckTimeout(now)$;
$temp = IntraZT_x\_CheckTimeout(now)$;
if $temp \neq 0$ **then**
  $B_x = (-1, x, deg(x), 0, temp)$;
end if
$NT_x = Update\_NT(B_{N_x})$;
$deg(x) = |NT_x|$;
$EPN_x = Determine\_PN(NT_x)$;
if $y \neq y'$ **then**
  if $PN(y) \neq x$ **then**
    $tmp = Remove\_IntraZT_x(y)$;
    $B_x = (-1, x, deg(x), 0, tmp)$;
  end if
end if
$y = y'$;
$Learned\_PN_x = Update\_IntraZT_x(B_{N_x}, y')$;
$Update\_InterZT_x(B_{N_x})$;
$ZID = Zone\_Name(Intra\_ZT_x)$;
$B_x = (ZID, x, deg(x), 1, y')$; //election modes
if $Learned\_PN_x \neq 0$ **then**
  $B_x = (ZID, x, deg(x), 0, Learned\_PN_x)$ // forward mode
end if

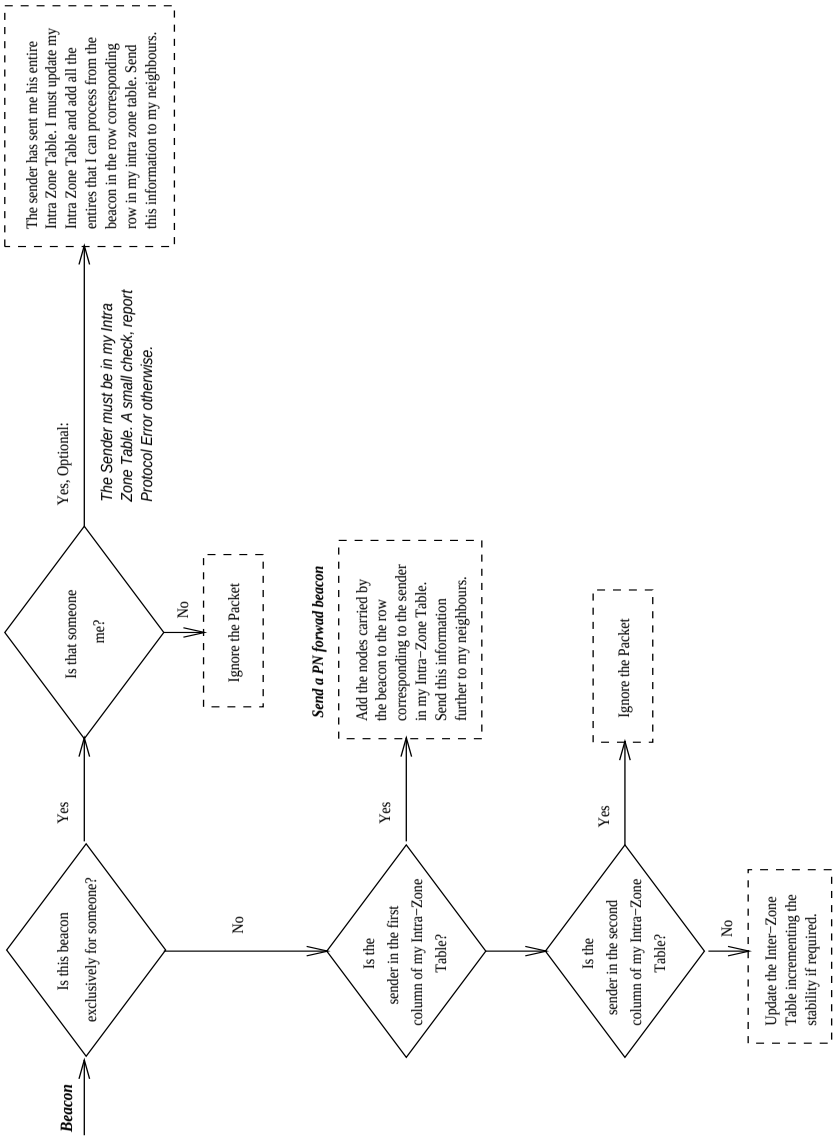


Fig. III-10. Flow chart of beacons in forward modes: a beacon arrives at a node x from a node y in PN Forward mode

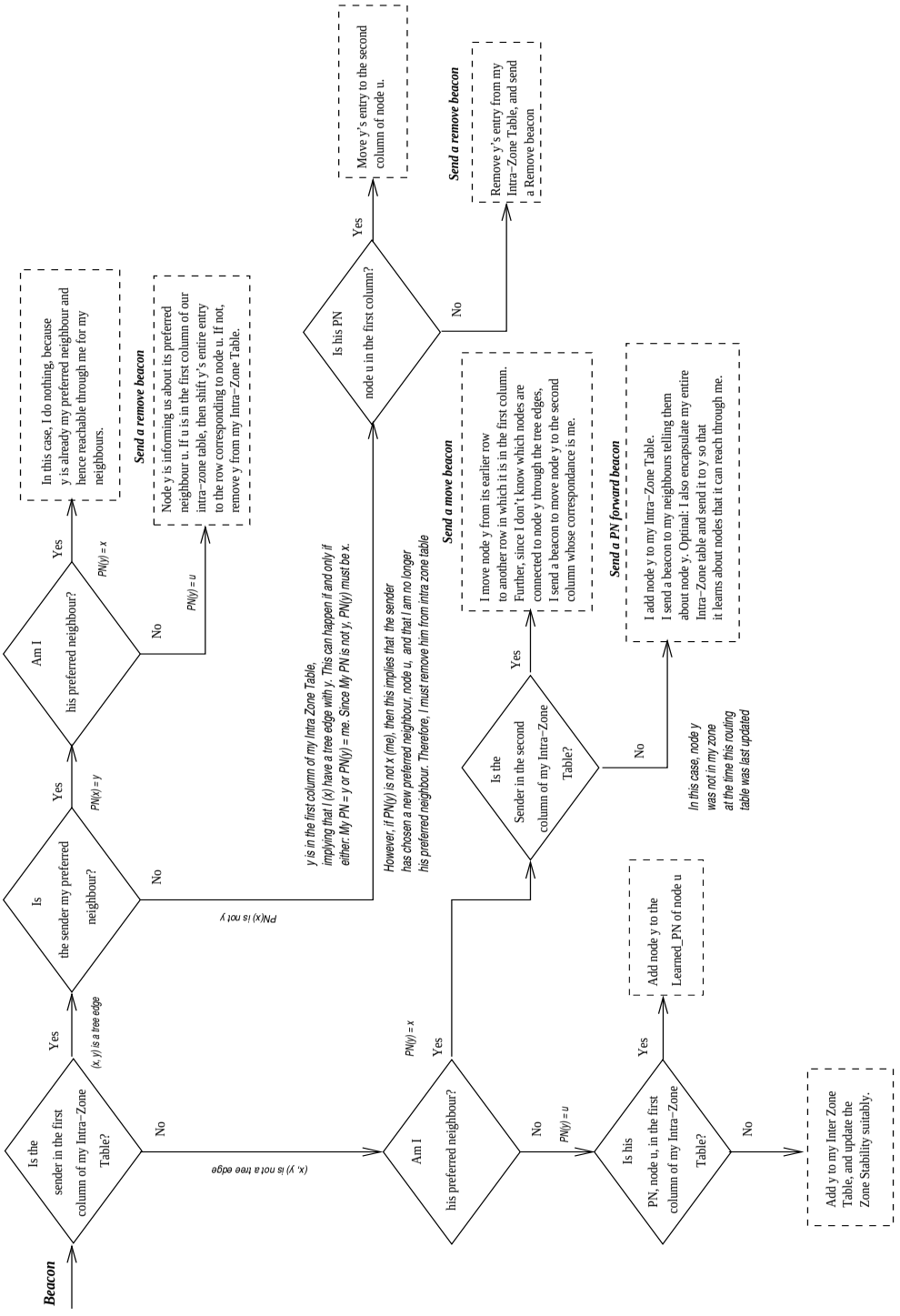


Fig. III-11. Flow chart of beacons in election modes: a beacon arrives at a node x from a node y in PN Elect mode

node generates $(d-1)$ beacons to forward the learned PN or removed PN, where d is the hop-wise zone diameter. In section I, we will find an upper bound for the zone diameter which is 8 hops. Therefore each zone generates $(d-1)N/M$ beacons. Since there exists M zones in the network, the overall generated forward beacons becomes $(d-1)N$. In conclusion, the total amount of communication overhead for creating topology CT produced by DDR is

$$S_{DDR} = pN + \tau(d-1)N \; messages, \; where \; d < 9 \qquad \text{(III-7)}$$

It can be shown that the complexity of DDR is always smaller than LSR, OLSR, and ZHLS. In OLSR, R_N is proportional to the number of nodes, whereas in DDR, d initially grows with number of nodes and stabilizes after certain threshold (see section I). Similar to LSR, in ZHLS the complexity is of the order of 2 while in DDR is of the order 1.

H NEW CRITERIA FOR PREFERRED NEIGHBOR ELECTION

As the degree of connectivity does not characterize the cause of link failure, we suggest a mechanism to describe the *quality* of connectivity for extracting the *links* connecting the pair of best nodes over time from the network point of view, and use this quality as *the* criteria for preferred neighbor election. This is because the performance of ad hoc routing strictly depends on the quality of each individual node. Indeed, this quality should not only reflect the available resources but also the stability of such resources; because first mobile ad hoc networks potentially have less resources than wired networks, and second mobility may result in link failure which in turn may result in a broken path. Therefore, more criteria are required in order to capture the quality of the links between nodes. We identify three metrics to represent the quality of connectivity from the network point of view: power lifetime, available unallocated buffer, and stability level. By changing the criteria of preferred neighbor election from degree of connectivity to quality of connectivity, we construct a forest of nodes with the high quality links. This is desirable because the subset of the set of forwarding nodes belongs to the nodes with the high quality, which in turn improves routing performance. This constructed forest is then dynamic and adaptive because the quality of nodes change over time according to the network conditions, i.e. network load, mobility, and available resources.

H.1 Quality of Connectivity at the Network Layer

We define *quality of connectivity* (QoC) as the *power level*, *buffer level*, and *stability level* of a node to represent the "quality" of nodes [57, 55]. For the sake of simplicity, we only consider *symmetric* environment where all nodes have similar capabilities

such as transmission range and buffer capacity. Note that the quality of connectivity is internal to a node and it is periodically evaluated by each node. The quality of connectivity of a particular node reveals whether the node is *forced* to be *selfi sh* or not. In the *selfi sh* mode, a node ceases to be a router and acts only as a host. We assume that each node periodically broadcasts its presence and its QoC in the form of a *beacon* to its neighboring nodes.

- ***Power Level*—** the power level represents the current battery life time. It represents a node's internal state, and we assume that a node is capable of determining its state. It is translated into a two-bit code that indicates the QoC of a node in terms of battery life time. We classify the QoC in terms of battery life time into *high*, *medium*, *low* and *selfi sh* states corresponding to each of the four two-bit codes. For example, a *high* QoC may be indicated if the battery life time is between 75% and 100% of its actual life time, and a node may exhibit *selfi sh* behavior in case its life time is lower than 25%. Intermediate battery levels may be classified into *medium* and *low* states.

- ***Buffer Level*—** which stands for the available unallocated buffer. Note that if the buffer level of a particular node is low, then this implies that a large number of packets are queued up for forwarding, which in turn implies that a packet routed through this node would have to experience high queuing delays. This metric is translated into a two-bit code which indicates the QoC of a node in terms of available buffer. This two-bit code is used to indicate *high*, *medium*, *low* and *selfi sh* QoC in terms of the buffer level. A *high* QoC indicates that the corresponding node no packets queued up for forwarding, while *selfi sh* QoC shows that the available buffer is less than 25 percent of its size. Since there is a slight delay between the broadcast of this metric and its use, instantaneous buffer-level may be misleading. Hence, a node should maintain an average buffer-level such as the exponentially weighted moving average (EWMA).

- ***Stability Level*—** we define the connectivity variance of a node with respect to its neighboring nodes over time as the stability of that node. This metric is used to avoid unstable nodes to relay packets. We estimate the stability of a node x as:

$$stab(x) = \frac{|N_{t_0} \cap N_{t_1}|}{|N_{t_0} \cup N_{t_1}|} \tag{III-8}$$

  N_{t_1} and N_{t_0} represent the nodes in the neighborhood of x at times t_1 and t_0 respectively. Note that, $t_1 - t_0$ denotes the time period in which nodes exchange beacons. A node is unstable if a large number of its neighbors change. Further, if most (or all) of the neighbors remain the same at two times t_1 and t_0, then we call this node stable. Note that $N_{t_1} \cap N_{t_0}$ (the numerator of $stab(x)$) denotes the set of nodes that have remained in the neighborhood of x between times t_0 and t_1. The denominator of $stab(x)$ is a normalization term. A node has *high* stability if none of its neighbors change ($N_{t_1} = N_{t_0}$) , in this case we have $stab(x) = 1$. A node

is *unstable* (no stability), if all its neighbors change ($N_{t_1} \cap N_{t_0} = \phi$), in this case we have $stab(x) = 0$. We say that a node has *low* stability if $0 < stab(x) \leq 0.5$ and that it has *medium* stability if $0.5 < stab(x) < 1$. A two-bit code maps the stability to four QoC of *high*, *medium*, *low* and *no* stability. For the sake of conformity with the other metrics, if a node has *no* stability, we say that it has *selfi sh* stability.

In order to facilitate the notion of QoC, we need to map the QoC onto a single weighted metric which can be compared and whose best can be chosen. Suppose s, b, and p denote the stability, buffer and power levels of a particular node. Note that $s, b, p \in \{0, 3\}$ since we are using a two bit code to capture these metrics. One way to estimate the QoC of node x is

$$QoC(x) = f(s, b, p) = \alpha \cdot s + \beta \cdot b + \gamma p \tag{III-9}$$

The weights α, β, and γ denote the relative importance of stability, buffer, and power level amongst themselves. For the sake of simplicity, we only consider stability and buffer level to determine the QoC of a node. Since we desire stability to be the most important followed by the buffer level. we propose $\alpha = 2$ and $\beta = 1$. Hence, given two nodes, we are always in a position to select the *better* one. For example, if node f has $s = 3$, $b = 3$ then its QoC is: 9. On the other hand node k with $s = 3$, $b = 2$ has a QoC value of 8. Hence, in our scheme, node f is a "better" node than node k.

H.2 Forest of High Quality Nodes

By changing the criteria of preferred neighbor election from degree of connectivity to the quality of connectivity, each node selects a neighbor whose QoC is the maximum in the neighborhood. Therefore, each node determines the set of high quality PN in the neighborhood as $PN_x = \{y | QoC(y) = \max(QoC(N_x))\}$, where N_x is the neighboring nodes of node x. Similarly, the forest is constructed by connecting each node to its preferred neighbor. This forest is dynamic and adaptive because the quality of connectivity change over time according to the network traffic load (affect the buffer level), network mobility (affect the stability level), and node available resources (affect the buffer and the power level).

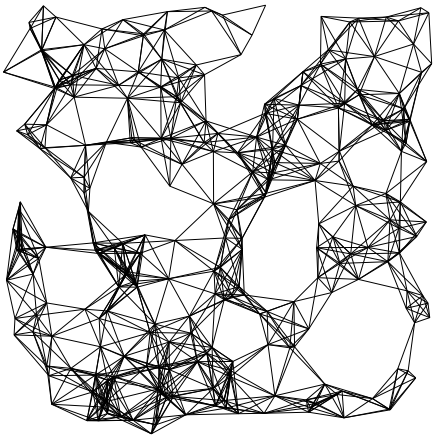
I BEHAVIORAL RESULTS ON TOPOLOGY MANAGEMENT STRATEGY

We have studied the behavior of the DDR algorithm with parameters such as zone diameter, average number of nodes in a zone, average number of zones in the network,

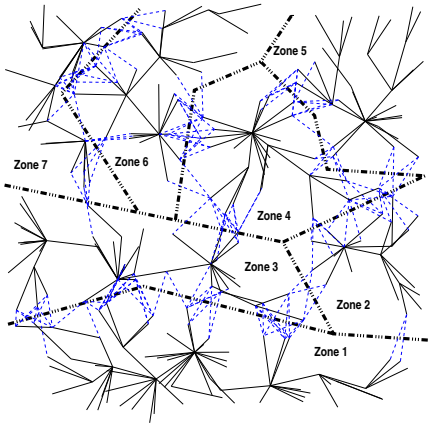
and the ratio of path length to the optimal length under various network density. The following results were obtained by implementing the static DDR algorithm for forest construction and for zone partitioning in C++ and measuring the metrics after the population of mobile nodes was distributed uniformly on a grid of $2000m \times 2000m$ with each node having a transmission range of 250m. The key aspect of these measurements is that they depict how DDR behaves with an increasing number of nodes in the network. We consider two cases: *variable density* where the network is sparse at the beginning and becomes highly dense, and *constant density* where the area covered by the ad hoc network increases as the number of nodes increases. The problems of desirable number of neighbors and the optimal transmission range to have a connected (or k-connected) network were addressed in [63, 64, 65, 66].

Fig. III-12(b) shows the forest constructed by DDR from an arbitrary graph of 250 nodes illustrated in Fig. III-12(a). In this figure, the solid edges represent the tree edges, the dashed edges denote the bridges, and the dash-doted lines show the zone partitioning. There are 7 zones $(1-7)$. For example zone 2 is connected to zone $1, 3, 4$ and 5 via several gateway nodes. So, if a node wishes to communicate with another node that is not in its zone, then a route discovery procedure is *only* performed along tree edges and bridges. Remember that we call the nodes that form the bridge edges gateway nodes. Note that the DDR algorithm does not necessarily select the min-hop path within a zone. The graphs in Fig. III-12(c) & III-12(d) show the number of edges in the network topology and the number of edges in the forest versus the number of nodes. There is a theoretical fact that the number of edges in a tree is of the order of the number of nodes. For a tree with n nodes, the number of edges is $n-1$. Further, a forest has strictly lesser edges than a tree spanning all the nodes. Therefore, the number of edges in the forest as reflected by the figures is $O(n)$. This graph reflects to what extent the overhead of flooding can be reduced if the forest structure is used. The algorithm attempts to dynamically adapt the network topology by extracting the high quality links from the network for the purpose of routing, and eliminating nodes with poor quality. Indeed, the algorithm detects low quality links, and adjust the network topology accordingly so as to reduce the probability of link failure.

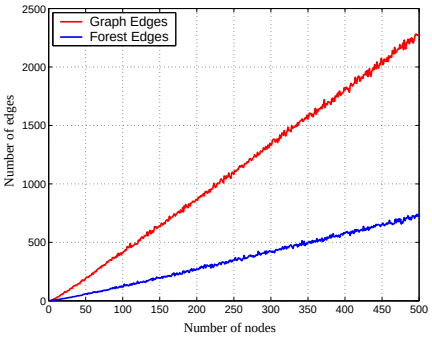
The graph in Fig. III-13(a) shows the number of zones versus the number of nodes in the forest generated by DDR for a constant and a variable density zone partitioning. In a constant density zone partitioning method, the number of zones increases linearly as the number of nodes in the network increases. This means that the communication overhead within a zone remains constant as the number of nodes in the network increases at the expense of an extra overhead for the communication outside of the zone. However, the average number of nodes in a zone stays constant as it can be noticed in Fig. III-13(b). In a variable density case, we observe that the more sparse the network is (below 200 nodes on a grid of 2000m $\times$ 2000m), the more partitioned the network becomes and hence the more number of zones the algorithm produces. As we increase the network density (from 200 up to 500) with the same size of the network, the number of zones remains constant and it tends to 7. This indicates that increasing network density has no effect on the number of zones but on the number of nodes in each zone as shown in Fig. III-13(b). Therefore, in contrast to the former case, the communica-



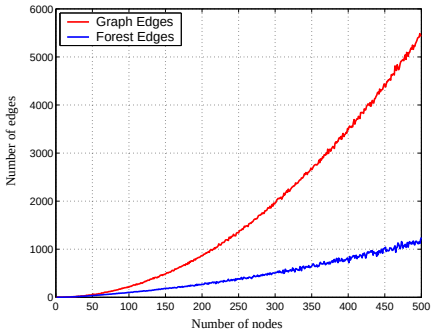
(a) An arbitrary graph



(b) The forest derived from the graph

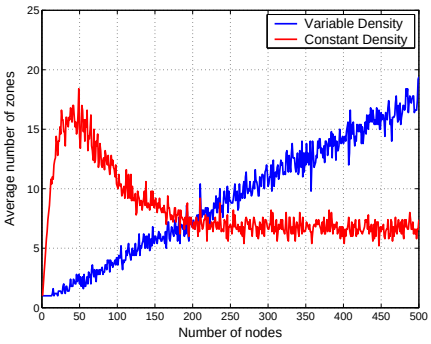


(c) Number of edges used for routing in the constant density case

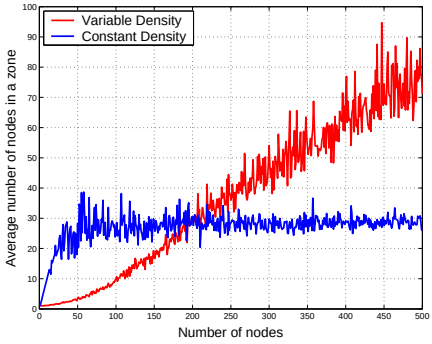


(d) Number of edges used for routing in the variable density case

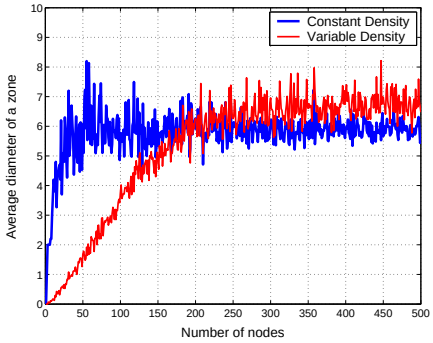
Fig. III-12. Number of edges used for routing as a function of number of nodes in a flat network and in the forest



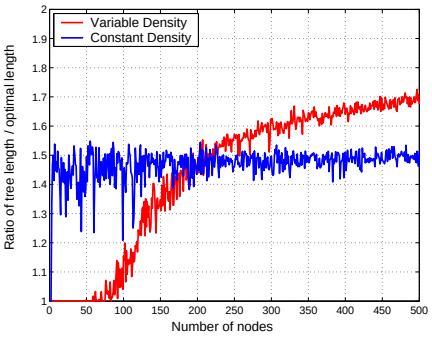
(a) The number of zones versus the number of nodes



(b) The size of zones versus the number of nodes



(c) The diameter of a zone versus the number of nodes



(d) The average ratio of the DDR hop count to the optimal hop count vs the number of nodes

Fig. III-13. Comparison of constant and variable density of zone partitioning methods

tion overhead within zones linearly increases as the number of nodes increases but the communication overhead outside of zones remains constant.

The graph in Fig. III-13(c) shows the diameter of a zone versus the number of nodes in the network. As stated before, the diameter of a zone is defined as the length of the path that has the longest hop count. If the zone diameter is fixed, then we can place an upper bound on the end to end delay for a connection between two nodes belonging to the same zone. For this reason, it is preferable to maintain the zone diameter as low as possible even if the number of nodes increases. From Fig. III-13(c) it can be inferred that even though the zone diameter increases initially as the number of nodes increases, it stabilizes for number of nodes greater than 200. From another point of view, the zone diameter determines the degree of trade-off between routing overhead and delay. The longer the zone diameter becomes, the more proactive the protocol is and vice versa. As a result, when the zone diameter increases, the route acquisition delay within the zone decreases (it tends to zero) but the overhead of the zone increases.

The graph in Fig. III-13(d) measures the ratio of the tree hop count length to the optimal (shortest) hop count. This measurement is important since if the hop count of source, destination pairs in a forest is higher than that of the shortest hop path, then our protocol may suffer from consuming high network resources and incurring high delays. The ratio is measured using an average. That is to say that the measurement is averaged over all possible source, destination nodes belonging to the same zone. Hence, if s and d are two mobile nodes belonging to the same zone in the tree T_i, and *Tree_Length* (s, d) denotes the number of hops on the path between s and d using the tree edges, and *Optimal_Length* (s, d) denotes the shortest number of hops, then the average measures:

$$\frac{\sum_{s,d \in T_i} \frac{\textit{Tree\_Length}(s,d)}{\textit{Optimal\_Length}(s,d)}}{\sum_{s,d \in T_i} 1} \qquad \text{(III-10)}$$

As is observed from Fig. III-13(d), we can deduce that on the average, the tree-length is no worse than twice the optimal shortest hop path for up to 400 mobile nodes in the topology. Furthermore, the trend of the graph suggests that this ratio is stable. This is a desirable result because the construction of the forest does not consider optimal hop count as a route determination metric to generate the forest.

J PROTOCOL MODEL

In order to study the effect of our topology management strategy on routing protocols, we use a hybrid routing protocol denoted as Hybrid Ad Hoc Routing Protocol (HARP). HARP combines a proactive behavior within a zone with a reactive behavior between zones. In fact, routing is performed on two levels: intra-zone and inter-zone, depending on whether the destination belongs to the same zone as the forwarding node [56]. Intra-zone routing relies on the proactive mechanism of our topology management strategy

DDR. It is important to note that intra-zone routing does not have any route acquisition delay thanks to the zone partitioning algorithm of DDR. Inter-zone routing, on the other hand, applies the path discovery procedure to find the shortest path to the destination's zone. In inter-zone routing, we try to conceal the zone details and to look upon them as nodes. The aim here is to establish a connection from the source node's zone to the destination node's zone. As stated before, a link that connects nodes belonging to different zones is called a *bridge*. Hence, in inter-zone routing, we are looking for a path of bridge edges that connect the zone of the source node to the zone of the destination node. During the process of path discovery, intermediate zones/nodes record the routing information in their routing table so as to establish the *reverse and forward path*. HARP will be explained in detail in chapter IV.

K SIMULATION MODEL

We use a simulation model based on *ns-2* in our performance evaluation. *ns* is a discrete event simulator developed by the university of California at Berkeley and the VINT project [67]. Recently, this simulator has been extended by the Monarch research group for supporting simulations in wireless mobile ad hoc environments. A comparison of *ns* with other popular simulators such as OPNET and GloMoSim (QualNet) can be found in [68, 69].

Our protocol maintains a transmission buffer of 64 packets. It contains all data packets waiting to be transmitted including packets for which route discovery has been started, but no reply has arrived yet. In order to prevent indefinite buffering of packets, packets are dropped if they wait in the transmission buffer for more that 10 simulated seconds. The beaconing period is 10 simulated seconds. We use traffic and mobility models similar to those previously reported using the same simulator [17, 70, 71]. Traffic sources are CBR (constant bit rate). The packet size is 512 bytes and the packet rate is 4 *packets/second*. The mobility model uses the *random way point* model in a rectangular field of $1500m \times 300m$. This model was first used by Johnson and Maltz in the evaluation of DSR [35], and was later refined by the same research group. In this model, each node begins the simulation by remaining stationary for a pause time. It then selects a random destination in the $1500m \times 300m$ space and moves to that destination at a random speed uniformly chosen from $(0, V_{max}]$, where V_{max} is the maximum speed of the simulation. This model is expected to maintain this average speed as the simulation progresses. Simulations are run for 900 simulated seconds for 50 nodes. The selected pause times, which affect the relative speeds of the mobile, are $0, 30, 60, 150, 300, 600,$ and 900 seconds. For each pause time, we randomly generate 10 different mobility scenarios. So, each data point in the performance results represents an average of 10 runs.

L PERFORMANCE RESULTS

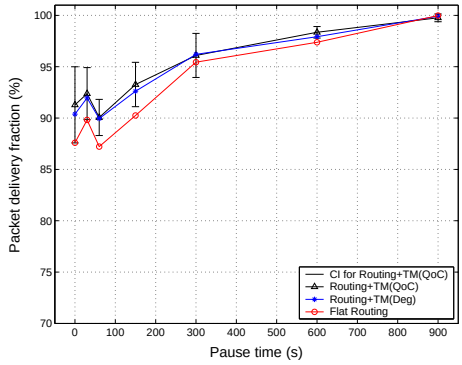
In the following, we compare the performance of the flat version of our routing protocol HARP (without any topology management strategy), with the ones that employ the topology management strategy. In the latter case, we vary the criteria for preferred neighbor election from degree of connectivity to quality of connectivity. In the following figures, these cases are distinguished by: *Flat Routing, Routing+TM(Deg)*, and *Routing+TM(QoC)* with its 95% confidence interval *CI for Routing+TM(QoC)* to differentiate each case. Three key performance metrics including *packet delivery fraction*, *average end-to end delay*, and *routing overhead* are evaluated under various traffic load, and various mobility rate.

L.1 Packet Delivery Fraction

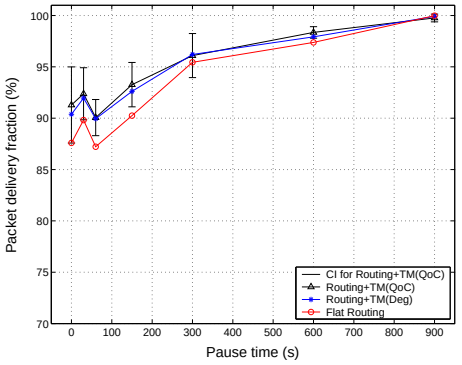
Packet delivery fraction is defined as the ratio of the data packets delivered to the destination to those generated by the CBR sources. Fig. III-14 compares this metric for the *Flat Routing, Routing+TM(Deg)*, and *Routing+TM(QoC)*. In the low traffic load (10 sources), the *Flat Routing* has a slightly better packet delivery fraction than the two others (Fig. III-14(a)). This is due to the shorter path length used in the *Flat Routing* and the fact that the network is not congested. Note that, in two other cases the average path length is no larger than 2 times of the optimal path length (c.f. section I). However, the *Routing+TM* strategies outperform the *Flat Routing* by about 7 percent at lower pause time (high mobility) for the medium traffic load, and up to 20 percent as the pause time decreases for the high traffic load. This is because routing based on the minimum hop count metric always biases the same class of routes. As a results those routes become congested as the traffic load increases in the network. A similar phenomenon was also observed in [72]. However in the *Routing+TM*, this bias is reduced due to the dynamic nature of the forest and hybrid behavior of routing leading to the load balancing in the network (Fig. III-14(b) & III-14(c)). Furthermore, the *Routing+TM (QoC)* outperforms the *Routing+TM (Deg)* in the high traffic load. This is because of the high quality links used during the routing process. Note that the high quality links (or nodes) reduce the probability of link failure specially in the high mobility, and hence improve routing performance.

L.2 Average end-to-end delay of data packets

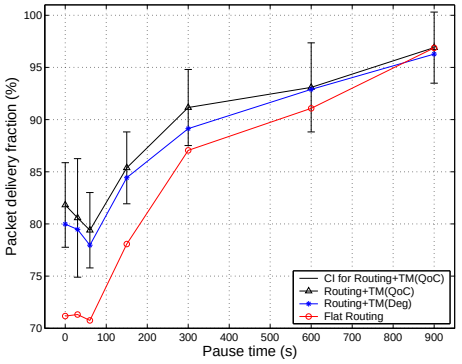
The average end-to-end delay includes all possible delays caused by buffering during route discovery latency, queuing at the interface queue, retransmission delays at the MAC, propagation and transfer times. As it is illustrated in Fig. III-15, the *Routing+TM* approaches have *always* lower delay than *Flat Routing*. Indeed with medium and high traffic load, they improve the delay performance up to 50 percent for lower pause times. Again, this is due to the hybrid behavior of routing and the dynamic na-



(a) 10 sources



(b) 20 sources



(c) 30 sources

Fig. III-14. Packet delivery fraction for the 50 nodes with various number of sources

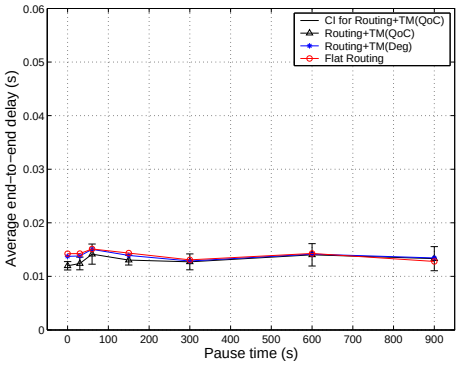
ture of the forest. Furthermore, *Routing+TM (QoC)* has even a better end-to-end delay since all forwarding nodes maintain high quality links. One interesting observation is that the delay increases with high traffic load with very low mobility rate. This is due to a high level of congestion and multiple access interfaces at certain regions of the network. Therefore, there is a need for a *load balancing* mechanism to distribute evenly the traffic when the network is congested. This phenomenon is less visible with higher mobility where traffic automatically gets more evenly distributed due to source movements. Note that both degree and quality of connectivity criteria for forest construction vary as a function of node mobility. Recall that in case of *TM (QoC*, we biased the weight of stability by $\alpha = 2$ against the buffer by $\beta = 1$ (c.f. equation III-9). A similar observation was also reported in [71, 70].

L.3 Routing overhead

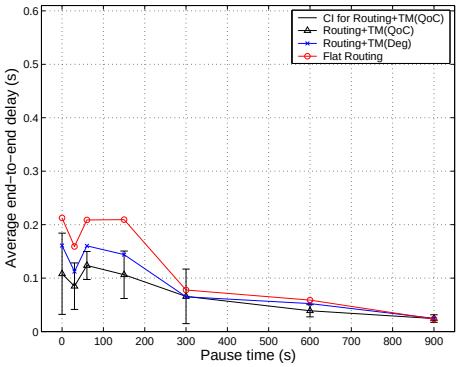
Routing overhead is measured as total number of bytes and packets used for routing process during the simulation. Only overhead stemming from the IP layer is included. The packet and byte overhead are shown in Fig. III-16 & Fig. III-17. Both *Routing+TM* protocols (Deg and QoC) have higher overhead than *Flat Routing* in low traffic load. This is due to the beaconing process used in these two approaches. Indeed, this process attempts to detect and react to link failures before its occurrence by adjusting the network topology specially in the *Routing+TM (QoC)* protocol. However in medium and high traffic load, the overhead for *Routing+TM* and *Flat Routing* are very similar specially in lower pause time. The reason is that the forest structure reduces the broadcasting overhead by selecting a subset of the neighboring nodes for forwarding a packet. We have also examined the overhead of TM(Deg) algorithm in terms of number of transmitted packets, and average transmitted data over the wireless interface under various mobility rate where no data traffic is present in the network (Fig. III-17(d) & III-16(d)). Obviously *TM(QoC)* generates more packets at the situation in the network becomes more stressful (c.f. equation III-9). From the results we have obtained so far, we deduce that the *Routing+TM* significantly improves the routing performance in comparison with the flat routing for all network conditions.

M ARCHITECTURE

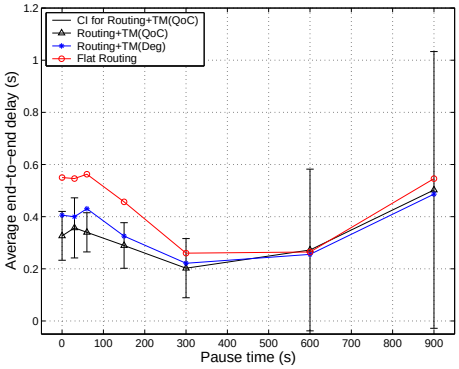
The main objective of our topology management strategy is to offer a flexible infrastructure on which several reactive routing protocols can be used according to desired requirements. Therefore, the reactive behavior between zones can be performed by any standard reactive topology-based routing protocols such as DSR [35], AODV [34], or RDMAR [40]; as well as by any reactive position-based routing such as TERMINODES [73], LAR [41], or GPSR [74]. These protocols can also implement the zone-level routing in addition to their standard node-level routing. Furthermore, the forest can be tuned by changing the *criteria* of the preferred neighbor election to achieve the desired requirements. For instance, we can set the criteria for the preferred neighbor



(a) 10 sources

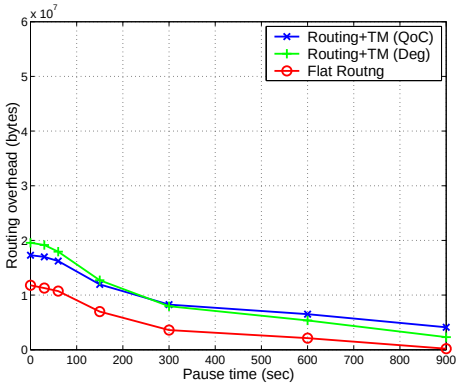


(b) 20 sources

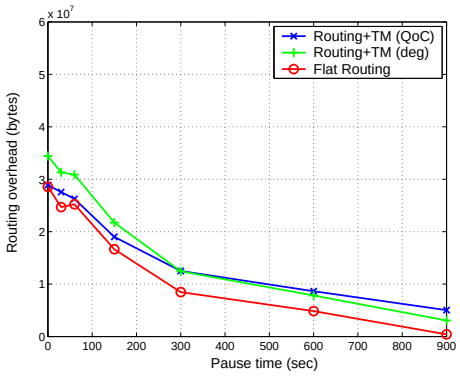


(c) 30 sources

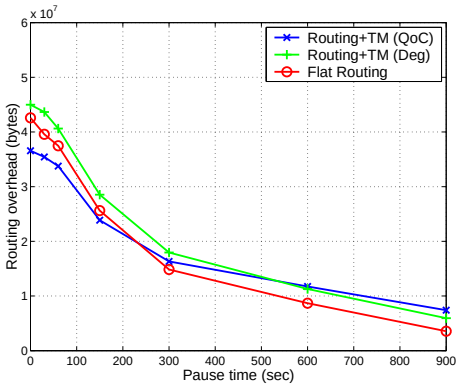
Fig. III-15. Average data packet delay for the 50 nodes with various number of sources



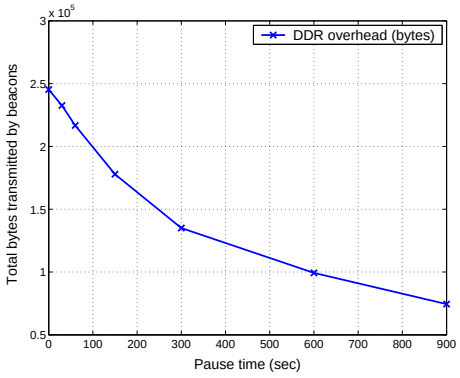
(a) 10 sources



(b) 20 sources

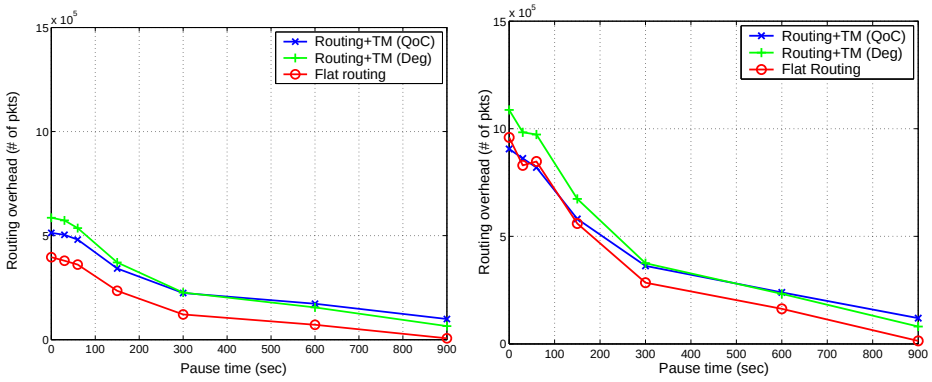


(c) 30 sources



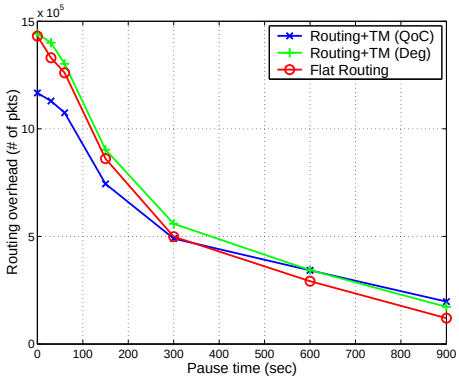
(d) Total transmitted bytes for TM (Deg)

Fig. III-16. Routing overhead in terms of number of packets and bytes for 50 nodes with various number of sources

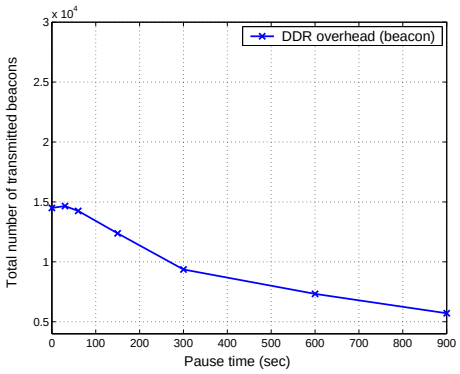


(a) 10 sources

(b) 20 sources



(c) 30 sources



(d) Number of transmitted beacon for TM (Deg)

Fig. III-17. Routing overhead in terms of number of packets for the 50 nodes with various number of sources

election to the minimum neighborhood degree instead of maximum neighborhood degree, which in turn provides other alternative routes to the destination node. These alternative routes can potentially balance the load of network. Moreover, some phases of the algorithm can be skipped regarding the requirements, for example the entire zone formation/construction or only the zone naming phase (see Fig. III-2).

Fig. III-18 represents the *architecture* of DDR and its interactions with routing protocols. As it is shown in the figure, the message from Mac layer is send to a classifier which is in charge of classifying the received message. A message can either be a beacon or a routing message. If beacon, then the classifier sends this message to the DDR box. Otherwise, the message is a routing message and has to be sent to the corresponding routing protocol. Notice that, each packet belonging to a routing protocol has its own header in order to be distinguished from packets of other routing protocols. DDR receives beacon messages from the classifier and sends beacon messages directly to the Mac layer without the intervention of routing protocols. DDR contains three tables: NB: neighboring table, ITA: intra-zone table, and ITE: inter-zone table. These tables have some local routing information that may be used by routing protocols to improve their performance. DDR provides an interface for the routing protocols to access these tables.

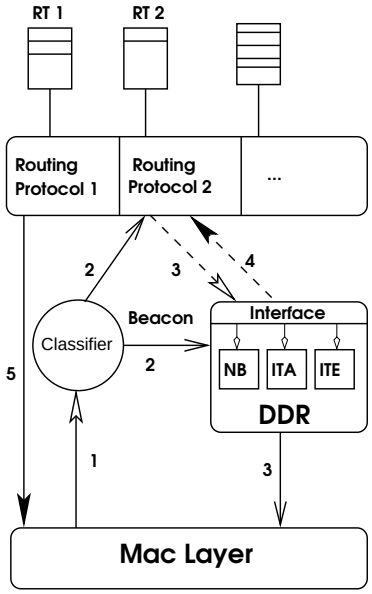


Fig. III-18. DDR Architecture

N CONCLUSION

We have proposed a new architecture that separates topology management from routing protocol. Topology management is a process to adjust the network topology as nodes move and environment changes so as to achieve better performance; While routing protocol is a process of generating and selecting paths, and forwarding user traffic between source and destination. That is why we believe that mobility management is different from routing. This architecture optimizes routing performance according to two criteria: quality of network and application requirements. Topology management generates a logical structure with respect to the quality of network, and the routing protocol generates, selects, and maintains paths to satisfy application requirements.

In our algorithm, we have combined three main concepts: forest, zone, and quality of connectivity. Zones were used in order to reduce the delay due to routing process and to reach high scalability. Forest have reduced the broadcasting overhead by selecting a subset of the set of neighboring nodes for forwarding a packet. The quality of connectivity is used to extract the links connecting the pair of best nodes for the purpose of routing. Furthermore, by changing the criteria of preferred neighbor election from degree of connectivity to quality of connectivity, we have constructed a forest of nodes with high quality links, which in turn improved routing performance.

We have shown that the performance of routing can be improved with the help of topology management. In particular, the principle for this improvement stems from the separation we made between topology management and route determination due to their different natures and functionalities. We have observed that routing protocols require a load balancing mechanism in order to evenly distribute the data traffic in the network. This can be achieved through both topology management strategy and quality of service routing. We have noticed the misbehavior of both criteria of preferred neighbor election as they fail to provide load balancing in the network when the rate of mobility is low.

The work presented herein is the first series of simulation studies in this area. We aim to study the effect of different and possibly multiple criteria for forest construction in order to achieve the load balancing in all network conditions. Indeed, we intend to tune the criteria of preferred neighbor election so that the constructed forest always achieve an evenly distribution of the traffic load in the network.

CHAPTER IV

HARP—Hybrid Ad Hoc Routing Protocol

Contents

A Routing Procedures 61
A.1 Advance Routing 62
A.2 Intra-zone Routing 62
A.3 Inter-zone Routing 64
A.4 Path Maintenance 67
A.5 Query Localization Mechanism 68
A.6 Summary 71
B Simulation Model 71
C Performance Evaluation 73
C.1 Packet Delivery Fraction 73
C.2 Average end-to-end delay of data packets 74
C.3 Routing overhead 77
C.4 Discussion 77
C.5 Related Work 82
D Conclusion 83

Abstract— *A hybrid routing protocol which combines proactive behavior within a zone and reactive behavior between zones is proposed. Routing is performed on two phases: intra-zone and inter-zone, depending on whether the destination belongs to the same zone as the forwarding node. Intra-zone routing relies on an existing proactive mechanism inherited from the topology management strategy, and as a result routing is done without any route acquisition delay. Inter-zone routing, on the other hand, applies the path discovery procedure to find the shortest path to the destination's zone . The overhead related to flooding nature of this procedure is noticeably reduced due to the combination of forest and query localization techniques. The forest structure reduces the broadcast storm problem by selecting a subset of neighboring nodes to forward a packet, while query localization limits the scope of path discovery.*

A detailed simulation model is used to study the performance of our approach. We evaluate and compare the performance of our routing protocol with DSR, AODV, and OLSR routing protocols under various network load and mobility. The obtained results show that our hybrid protocol outperforms other protocols. It is demonstrated that our routing protocol together with the tuned topology management strategy succeed to achieve load balancing in the network. We also observe that network load and mobility affect the absolute performance of the protocols, and that their impact on different protocols is non-uniform.

Keywords— *Mobile ad hoc networks, routing, zone, flooding, query localization, simulation, performance evaluation.*

THE hybrid ad hoc routing protocol (HARP) is a simple and efficient routing protocol designed for mobile ad hoc environments [56, 75]. HARP maintains some short routes between nodes within a local neighborhood, and discovers long routes on-demand otherwise. Its goal is to achieve an optimal trade-off between routing overhead and delay while maximizing the network resource utilization. Using HARP, the network is fully autonomous and self-organized, and requires no fixed infrastructure. HARP functions at two levels: zone and node depending on whether the topology management is present. By the zone level routing, we mean to conceal the zone details and look upon them as nodes. Initially, each node knows its own ID; and after the network is established, each node knows the connectivity within its zone and the connectivity with the neighboring zones. The aim here is to establish a connection from the source node's zone to the destination node's zone. As stated before, a link that connects nodes belonging to different zones is called a bridge. In this case, we are looking for a path of bridge edges that connect the zone of the source node to the zone of the destination node. HARP avoids a single point of failure by enabling all gateway nodes to contribute in routing process.

HARP forms a new table called *routing table* from *intra-zone* and *inter-zone* tables because these two tables may change during the transmission even though the route between the source and the destination remains unchanged. When a source node receives a packets from its application layer, it first searches its routing table to check if an active route to the destination is present. If a valid route is found, the packets are routed accordingly. Otherwise, if no active route is available, a new entry will be created for this flow and then the routing process is launched. The routing process is performed in two phases: intra-zone and inter-zone, depending on whether the desti-

nation belongs to the same zone as the forwarding node. Intra-zone routing employs the intra-zone table in conjunction with routing table for data forwarding. As a result, routing is done without any route acquisition delay. Inter-zone routing, on the other hand, applies the path discovery procedure to the forest. Distance estimation mechanism is also invoked to find a distance from the source to the destination. The estimated distance is then converted to the hop count. The TTL field in the request message is set equal to the hop count before being broadcast by the source node. The combination of forest structure and query localization techniques achieves two goals at the same time: forest structure reduces the broadcast storm problem by selecting a subset of the neighboring nodes to forward a packet; while query localization technique limits the scope of the path discovery. This combination is desirable because it reduces not only the broadcasting overhead, but also the collision probability. Hence, it contributes to routing performance. There exists different design choices for the path discovery procedure used in the inter-zone routing. For instance, routing can either be *source based* like DSR [35] or hop-by-hop based like AODV [34]. In addition to this, the routing granularity can either be at the node level (e.g. AODV or DSR) or at the zone level (e.g. ZHLS [43]. If we combine source routing and zone level routing, the resulting routing will only add the zone ID to the route record of the packet. Each zone ID may go along with the gateway nodes if required. HARP applies a hop-by-hop routing both at the node and at the zone level (i.e. zone-by-zone routing), and therefore it inherits from both AODV and ZHLS. HARP currently utilizes symmetric links between neighboring nodes, and assumes that all nodes are willing to participate in the network protocols and packet forwarding.

This chapter describes the design of the HARP protocol and its interaction with topology management strategy, and provides its performance comparison under various traffic load and mobility rate. Section A presents different design phases of the protocol, in particular we describe the design of intra- and inter-zone routing, path maintenance procedure, followed by the query localization technique. Section B highlights the simulation model for our performance evaluation. In section C, various simulation results are presented. These results point out that the design goals for providing an optimal trade-off between routing overhead and delay, while still offering efficient route establishment have been achieved. Finally, we summarize our protocol and its behavior under different network conditions, and provide some directions for the future work.

A ROUTING PROCEDURES

The HARP protocol is composed of four mechanisms that work together to allow determination and maintenance of paths in the ad hoc networks: *advance routing, intra-zone routing, inter-zone routing,* and *query localization*. Advance routing benefits from the existing valid route in the routing table for packet forwarding. Intra-zone routing uses the intra-zone table which follows the forest structure except for when the destination is in the neighborhood of the forwarding node. The inter-zone routing initiates the path discovery process whenever a source node requires to communicate with another

node for which it has no valid routing information in its tables. In this process, a path request message is broadcast throughout the network to reach the destination. Query localization estimates the relative distance between the source and destination nodes before starting path discovery. The estimate is computed on the basis of the previously known value of the distance between the source and destination nodes and the time elapsed since this value was recorded. The following sections further elaborate these mechanisms.

A.1 Advance Routing

HARP develops a new table called *routing table* from intra-zone and inter-zone tables because these two tables may change during the transmission even though the route between the source and the destination remains unchanged. Note that changes in the zone tables are due to dynamic nature of the forest caused by preferred neighbor election algorithm (c.f. Chapter III). Table IV-1 shows an entry of the routing table which consists of 5 fields. The first field includes source address [src@], destination address [dst@], and port number [port#]. It is used to identify different flows. The second field represents the last and the next zone or hop in the path from source to destination. This field is used to reach both the destination and the source node if required. Note that the field [last,next] may change according to the level of routing. The third field [metric] shows the desired quality of service required by the flow. The next field [up, down] is used to keep the distance of a node from the source and from the destination. This field will be used to estimate the relative distance between the source and destination nodes on the basis of the previously known distance and the time elapsed since this value is recorded. The last field represents the last updated time for the entry, and it is used to purge the stale entries after a predefined threshold.

TABLE IV-1. ROUTING TABLE ENTRY

[src@, dst@, port#]	[last, next]	[metric]	[up, down]	[LUT]

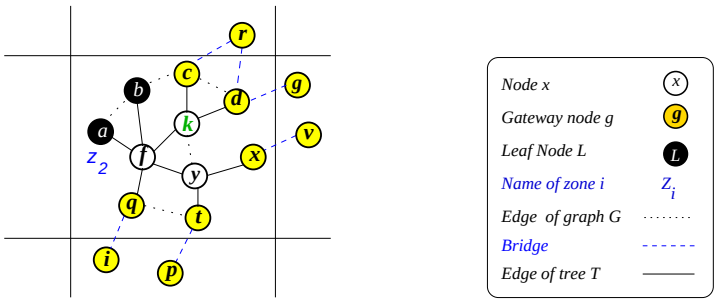
When a source node receives a packet from its application layer, it first searches its routing table to check if an active route to the destination is present. If a valid route is found, the packet is routed accordingly. Otherwise, if no active route is available, a new entry will be created for this flow and then the intra-zone and inter-zone routing process are launched.

A.2 Intra-zone Routing

Intra-zone routing is proactive in nature, and uses the intra-zone table to forward the data traffic. Indeed, it follows the structure provided by the tree except for when the

destination is in the neighborhood of the forwarding node. Assume that node x wants to forward data to destination node d. Before sending data to the node d, node x checks if node d exists in its neighboring table or intra-zone table. The former case is straightforward. In the latter case, node x creates a new entry in its routing table and sets the field [last, next] accordingly. Note that the sub-field [next] is obtained according to x's intra-zone table. Then, node x *forwards* the data to the next hop towards node d. Pseudocode 2 outlines the advance and intra-zone routing mechanisms, and show how inter-zone routing is triggered.

For example in Fig. IV-1(a), consider the scenario where node k wants to communicate with one of the nodes within its zone z_2, e.g. f, b, q, y, c, d, x, t. According to its intra-zone (see Table IV-2) and neighboring tables (see Fig. IV-1(a)), k can reach x, t, q, a, b via f, while other nodes f, c, d, y are directly reachable. Therefore intra-zone routing table always indicates the next hop for each destination within the zone. Let's consider another scenario where node c wants to send data packets to y, x or r, and identify the paths. In this scenario, the path connecting node c to node y is $p_{(c,y)} =< c, k, y >$, to node x is $p_{(c,x)} =< c, k, f, y, x >$, and to node r is $p_{(c,r)} =< c, r >$ (see Table IV-2 & Fig. IV-1(a)).



(a) Example (b) Legend

Fig. IV-1. Intra-zone routing

TABLE IV-2. INTRA-ZONE TABLE OF NODES k, f AND c REGARDING FIG. IV-2(A)

NID	Learned_PN
f	a, b, q, y, t, x
c	-
d	-

(a) $Intra\_ZT_k$

NID	Learned_PN
y	x, t
k	c, d
b, a, q	-

(b) $Intra\_ZT_f$

NID	Learned_PN
k	f,d,a, b, q, y, t, x
-	-
-	-

(c) $Intra\_ZT_c$

Pseudocode 2 Routing procedures

Let $WAITING\_FOR\_PREP$ be the time that the source waits to receive $PREP$.
RD is a function that estimates the relative distance between S & D (see section A.5).
if $dst \in Routing\_Table$ **then**
  $next\_hop = determin\_next\_hop(dst)$;
  if $next\_hop \neq waiting\_for\_reply$ **then**
    $forward(data, next\_hop)$;
  end if
else
  $create\_routing\_table\_entry(src, dst, port, last, next, metric, down, up, lut)$;
  if $dst \in my\_Intra\_ZT \,||\, dst \in my\_Neighboring\_Table$ **then**
    $next\_hop = determine\_next\_hop\,(dst)$;
    $reverse\_path\_setup(upstream\_node)$;
    $forward\_path\_setup\ (next\_hop)$;
    $forward(data, next\_hop)$;
  else
    $next\_hop = WAITING\_FOR\_PREP$;
    $inqueue(date)$;
    $TTL = RD(dst, my\_mobility, path\_mobility, elapsed\_time)$;
    $broadcastPREQ$;
  end if
end if

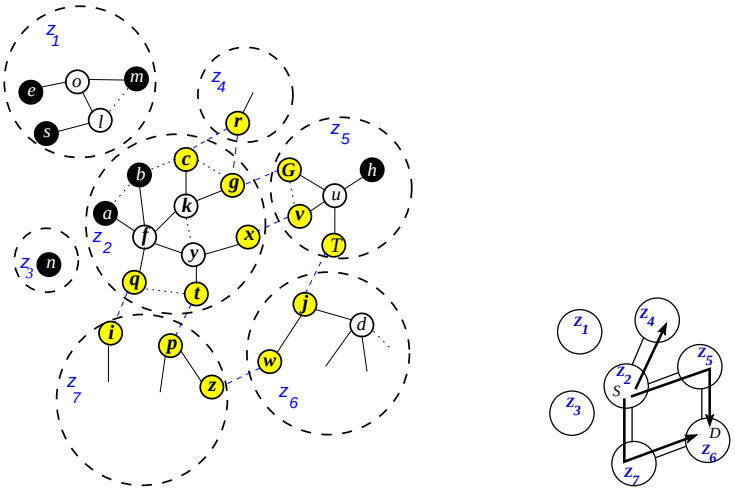
A.3 Inter-zone Routing

The inter-zone routing initiates the *path discovery* process whenever a source node requires to communicate with another node for which it has no valid routing information in its tables. The path discovery process consists of two parts: path request PREQ and path reply PREP. The path request propagates from zone to zone via the gateway nodes using both the intra-zone table and the inter-zone table; while the path reply is unicast from the destination back to the source. Each time a node initiates path discovery, it estimates the relative distance between the source and destination and sets the *time-to-live* field in the path request message to the converted hop-wise distance, which limits the scope of flooding. Within a zone, the path request message follows the tree structure, and as soon as a gateway node is found it traverses the zone. As a consequence, the path request propagation is limited to a subset of forwarding nodes. Each forwarding node applies the intra-zone routing as soon as it finds out that the destination node belongs to its zone. If additional copies of the same PREQ are later received, these packets are discarded. Hence, HARP only establishes loop free paths since it only propagates a *unique* PREQ on the *forest* (i.e. we have no cycle). During the process of forwarding the PREQ, intermediate zones/nodes record the routing information in their routing table (Table IV-1) so as to establish the *reverse path*. In this way, the node knows how to forward PREP to the source if one is received later. This entry will be deleted after the specified lifetime. If the PREQ is lost or not reached

to the destination (wrong distance estimation), the source node is allowed to retry the broadcast path discovery mechanism. Simulations have shown that the optimal number of retry is 2 [34].

After the limited flooding, several paths may be identified as potential candidates for a given destination. The destination chooses the shortest path by transmitting a path reply PREP back to the zone/node from which it received the PREQ. As the PREP is *unicast* back along the reverse path, nodes along this path set up a *forward path* in their routing table to the zones/nodes from which the PREP came. The forward path is then used to carry the source's data traffic to the destination. The forward and reverse path setup used in HARP are similar to the ones in AODV [34, 50]. Pseudocodes 3 and 4 outline the path discovery mechanism used in HARP.

Fig. IV-2(b) shows the zone level topology of our example in Fig. IV-2(a). For instance, node s from z_2 wants to communicate with node d from z_6. In this case, the PREQ is propagated in zones z_4, z_5 and z_7, and only forwarded in zone z_6 towards d. The solid arrow lines show how the path request PREQ message is propagated in the network.



(a) Example

(b) Inter-zone routing and path request propagation

Fig. IV-2. Inter-zone routing

Pseudocode 3 Path discovery procedure::Handle Path Request PREQ

```
Let WAITING_FOR_PREQ be the time the destination waits to receive PREQs.
if selfish then
    drop (PREQ); //optional
end if
if dst == my_address then
    update_list(PREQ);
    PREP = path_selection (PREQ);
    reverse_path_setup (PREP);
    schedule (PREP, WAITING_FOR_PREQ);
    return;
else if Not in_list(PREQ) then
    if sender ∈ my_Intra_ZT || sender ∈ my_Inter_ZT then
        update_list(PREQ);
    end if
    if Not leaf_node || dst ∈ my_Intra_ZT || dst ∈ my_Neighboring_Table
    then
        reverse_path_setup (PREQ);
        next_hop = determin_next_hop (dst);
        if next_hop ≠ unknown_dst then
            forward (PREQ, next_hop);
        else
            PREQ = update_path_request (PREQ);
            broadcast(PREQ);
        end if
    end if
else
    drop(PREQ);
end if
```

Pseudocode 4 Path discovery procedure::Handle Path Reply PREP

```
forward_path_setup (PREP);
if my_address == src then
    next_hop = determin_next_hop(dst);
    forward(data, next_hop);
else
    backward(PREP);
end if
```

A.4 Path Maintenance

Once the path to the destination is discovered, path maintenance procedure monitors the operation state of the path and informs the sender of any path error. If the source node moves, it reinitiates the routing procedure (i.e. intra-zone and inter-zone) to find a new path to the destination. If any node along the path moves, its upstream node notices the move and encounters two cases: either it backwards a *path error* message PERR to its upstream neighbor to inform the removal of the path by which the packet reached this host; or it performs a path discovery for the source and becomes the sender. The decision is made on the basis of the relative distance of the upstream node from the destination node. Supposing that node x notices that its neighboring node on the path from source node s to destination node d has moved. If the relative distance (of its downstream nodes along the path) from x to d is smaller than that of (its upstream nodes along the path) from s to x or the destination node belongs to the same zone as node a, then node a initiates a path discovery procedure. Otherwise, the path error PERR is backwarded to the upstream neighbor of node a, and so on until the source node is reached. When a source node receives the PERR, it may reinitiate a path discovery procedure if required. Pseudocode 5 outlines the path maintenance procedure.

An additional aspect of the path maintenance is the use of intra-zone routing if the intended destination belongs to the zone. An upstream node that encounters a link failure or receives a path error PERR message may use its intra-zone table to re-route the date packets.

Pseudocode 5 Path Maintenance Procedure

```
if can_reach (next_hop) then
  resend_data(data);
else
  RD_down = RDM(downstream, my_mobility, down_path_mobility, elapsed_time);
  RD_up = RDM(upstream, my_mobility, up_path_mobility, elapsed_time);
  if RD_down > RD_up || dst ∈ my_Intra_ZT then
    next_hop = waiting_for_reply;
    inqueue(date);
    TTL = rd;
    broadcastPREQ;
  else
    dropdata;
    backward(PERR);
  end if
end if
```

A.5 Query Localization Mechanism

Broadcasting in mobile ad hoc networks provides essential routing functionality for a number of routing protocols. Currently, these protocols rely on a simplistic form of broadcasting called *flooding*, in which each node retransmits each received unique packet exactly one time [45, 33]. One of the problems related to the flooding is the broadcast storm problem [47], where a node receives the same message from multiple neighboring nodes at about the same time. This potentially generates high overhead in the network. In our approach, this overhead is noticeably reduced by the forest structure since only a subset of neighboring nodes forwards the received unique packet. Another important issue is related to the *scope* of flooding, which in general is the diameter of the network. One approach to reduce the scope of flooding is the expanding ring search strategy which initially broadcasts the request to a small area and if the destination lies within that region, the query is successful [35, 49]. If it lies beyond the expected region, a retry is made with increased time-to-live (TTL). This method may perform even worse than flooding if the destination is at the edge of the network. HARP avoids flooding by using query localization mechanism to limit the scope of the discovery. This mechanism develops the idea of relative distance estimation (RDE) algorithm proposed by Aggelou and Tafazolli [40, 76], and provides some significant improvements to the algorithm. HARP attempts to discover routes quickly without incurring high control overhead with the help of query localization. Localization is achieved by estimating the relative distance between the source and destination nodes before starting route discovery. The estimate is computed on the basis of the previously known value of the distance between the source and destination nodes and the time elapsed since this value was recorded. The distance is then converted into hop count, which is then used to limit the search area and thus reduce control overhead.

HARP employs the routing table (c.f. Table IV-1) to obtain some information corresponding to the destination node about the previous relative distance (hop count) and the time when it was recorded. The time elapsed since last update can be easily determined. In the standard version of the algorithm, we take a unique moderate average velocity for each node, $micro\_velocity$, so as to calculate the offset for both the source and destination nodes [40]. The offset of a node is defined as the maximum distance the node may have with its original position if travels during a time interval. This assumption may fail to determine the accurate offset if various mobility rates are present in the network. Therefore, we use the notion of stability level introduced in Chapter III, in order to determine the *actual mobility rate* for our algorithm. In fact, a high level of stability indicates a low state of node mobility, while a low level may indicate a high state of node mobility. Furthermore, the offset is *only* calculated for the source and destination and not for the intermediate nodes, however, the mobility of intermediate nodes may cause a longer route even if the positions of source and destination remain unchanged. Therefore, it is essential to determine the actual mobility not only for the source node but also for the path down to the destination node. That is why we introduce two offsets: source and path, which are determined according to the source and path mobility rate for more accurate estimation of the distance.

In order to determine the path mobility, we define path stability, which is computed as a concave function of the stability level of each individual nodes (see Equation IV-1), and it is determined during the path discovery procedure. With the knowledge of node stability and path stability, we estimate the mobility rate for a node and for a path. Suppose P is a path between source node s and destination node d (that is P is a sequence of (non-repeating) nodes, $P =< s, n_1, \ldots, n_k, d >$). N_{t_1} and N_{t_0} represent the nodes in the neighborhood of x at times t_1 and t_0 respectively. We estimate the value of path stability using a concave function:

$$P.stability = \min_{n \in P \setminus \{s\}} n.stability, \quad where \tag{IV-1}$$

$$stability = \frac{|N_{t_0} \cap N_{t_1}|}{|N_{t_0} \cup N_{t_1}|} \tag{IV-2}$$

This value can then be mapped to the mobility rate. Recall from previous chapter that, we classify the stability into *high*, *medium*, *low* and *selfi sh* states. Therefore, the path mobility can be discovered by the nodes along the path connecting the source node to the destination node, and be used to determine the offset. The offset is the product of mobility rate and elapsed time; and it is calculated both for the source node, src_offset; and for the path, $path\_offset$. The path offset as its name indicates is calculated for the path (excluding the source), which is why it is multiplied by the path length d. It has to be mentioned that d is the old hop-wise length where the destination is found and not the old estimation.

$$src\_offset = source\_mobility\_rate \times elapsed\_time \tag{IV-3}$$

$$path\_offset = d \times path\_mobility\_rate \times elapsed\_time \tag{IV-4}$$

The new relative distance (RD), denoted as RD_offset, is then the sum of the above offsets plus the physical distance between source and destination (i.e. $d \times micro\_range$) divided by the average transmission range, $micro\_range$, to produce a normalized hop-wise distance.

$$RD_{normalized} = \lfloor \frac{RD\_offset}{micro\_range} \rfloor + 1, \quad where \tag{IV-5}$$

$$RD\_offset = source\_offset + path\_offset + d \times micro\_range \tag{IV-6}$$

The source node limits the scope of broadcasting by inserting the normalized value of RD in the *time-to-live*(TTL) field of path request PREQ message. The algorithm resets the TTL to the radii of the network whenever the elapsed time reaches a predefined threshold. This procedure is called relative distance microdiscovery (RDM) as in the

standard version [40]. The algorithm always expands the relative distance if the offsets are non-zero. Indeed, the way in which the new RD is estimated follows a monotonic increasing function depending on the mobility rate and the elapsed time. In Fig. IV-3, the two points marked ***src*** and ***dst*** denote the positions of the source and destination nodes respectively. The circle centered at ***src*** represents the maximum distance that the source node can move between the last update and current time. The same holds for the circle corresponding to the path. Thus, the two circles bound the new relative positions of the source and the path.

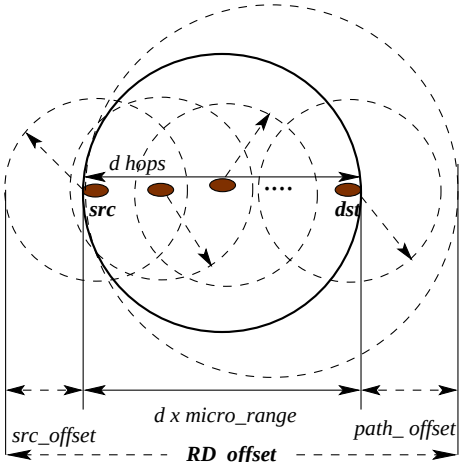


Fig. IV-3. Relative distance estimation

In the standard version of RDE, only the source node is allowed to estimate the RD and to set the TTL of packets. This has two side effects: (i) if the source node has no distance information about the destination node, then the TTL will be the diameter of the network; (ii) the path request message is propagated on the every direction of the source node for RD hops. To remove these side effects, we enable all the intermediate nodes to estimate the distance to the destination. For this purpose, we take the advantage of the [down] sub-field in the routing table. Recall that [up] indicates the number of hops to the *source* node, and is obtained during the path discovery procedure. Indeed when the source node sends a path request, all the nodes in its RD radius obtain the hop-wise distance to it and therefore they can estimate the distance to the source node if required. So, if the source node becomes the destination node, nodes in its previous RD radius can decide whether they are in the direction to that node or not. Indeed, each node encounters two main cases: either they are *not* in the right direction since they determine a larger valid estimation than the estimation made by the source and hence they drop the PREQ, or they are in the right direction and therefore they forward the PREQ. Moreover, nodes may have a smaller valid estimation than the current TTL, which also means that they are in the right direction, and thus they can update the TTL.

For example in Fig. IV-4, when s sends a PREQ, nodes x and y obtain the distance to s

by setting the [up] sub-field to the number of hops that this PREQ has been traversed so far. Therefore, if one of the nodes in the PREQ radius, say x, wants to communicated with s; it may use the value of [up] to determine the new RD, in order to broadcast a PREQ. Upon receiving x's PREQ, y compares its estimation with x's estimation. Node Y drops the request if it determines a larger estimation than x's estimation (upper case Y in the figure). Otherwise, y forwards the path request. If y's estimation is less than the value of TTL, it updates the TTL.

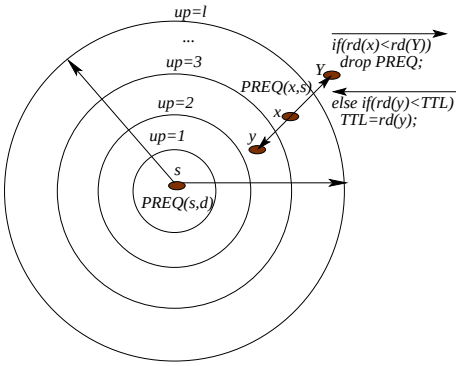


Fig. IV-4. Relative distance estimation

A.6 Summary

HARP does not attempt to maintain or discovers routes from every node to every other node in the network. It maintains some short routes to the destinations within a local neighborhood, while long routes are discovered on demand. The path discovery is not only localized but also directed towards the destination node. HARP performs routing on two phases: intra-zone and inter-zone depending on whether the destination node belong to the same zone as the forwarding node. Intra-zone routing is done at the node level, while inter-zone routing at the zone level. HARP is loop free at both the intra-zone and the inter-zone routing.

B SIMULATION MODEL

We use a simulation model based on *ns-2* in our performance evaluation. The simulation model uses 50 wireless nodes forming an ad hoc network. These nodes move over a rectangular ($1500m \times 300m$) flat space during 900 seconds of simulation time. We choose a rectangular space in order to use longer routes between nodes in comparison with square space. The physical radio characteristics of each mobile node's network interface, such as the antenna gain, transmit power, and receiver sensitivity, are chosen

to approximate the Lucent WaveLAN direct sequence spread spectrum radio characteristics [77]. The nominal bit-rate is 2 Mb/s and the nominal radio range is 250 meters. In order to make a fair performance comparison among protocols, it is critical to evaluate the protocols with identical loads and environmental conditions. Each simulation accepts as input a scenario file that describes:(i) the exact movement of each node; (i) the exact sequence of packets originated by each node; (iii) the exact time at which each change in movement or packet origination for each node occurs. We generate 210 different scenario files with varying movement patterns and traffic loads similar to those previously reported using the same simulator [17, 70, 71].

The movement scenario files used for each simulation are characterized by a *pause time*. Each node begins the simulation by remaining stationary for pause time seconds. It then selects a random destination in the $1500m \times 300m$ space and moves to that destination at a speed uniformly distributed between 0 and 20 meters per second. Upon reaching the destination, the node remains stationary again for pause time seconds, selects another destination, and moves towards the destination. It repeats this behavior for the duration of the simulation. This model was first used by Johnson and Maltz in the evaluation of DSR [35], and was later refined by the same research group. Recently, Yoon et al. have shown that the random way point model in its current form fails to reach a steady state in terms of instantaneous average node speed, but rather the speed continuously decreases as simulation progress [78]. This issue is still under consideration. Also, Camp et al. provide valuable simulation results that illustrate the importance of choosing a mobility model in the simulation of an ad hoc network protocol [9]. Movement patterns are generated for 7 different pause times: $0, 30, 60, 150, 300, 600,$ and 900 seconds. A pause time of 0 seconds corresponds to continuous movement, and a pause time of 900 (the length of the simulation) corresponds to no movement. Due to the impact of the moving pattern on the performance of the protocols, we generate scenario files with 70 different movement patterns, 10 for each value of pause time. All three routing protocols have been analyzed with these 70 different movement patterns.

Traffic sources are CBR (constant bit rate). The data rate is equal to 4 packets per second. Three different communication patterns are used corresponding to $10, 20,$ and 30 CBR sources with a packet size of 512 bytes. All communication patterns are peer-to-peer, and connections start at times uniformly distributed between 0 and 180 seconds. The three communication patterns ($10, 20,$ and 30 sources), taken in conjunction with the 70 movement patterns, provide a total of 210 different scenario files with which we compare the routing protocols. Our protocol maintains a transmission buffer of 64 packets. It contains all data packets waiting for a route such as packet for which route discovery has started, but no reply has arrived yet. To prevent buffering of packets indefinitely, packets are dropped if they wait in the transmission buffer for more that 10 simulated seconds. Beaconing period is set to 10 simulated seconds.

C PERFORMANCE EVALUATION

Three key performance metrics: *packet delivery ratio, average end-to end delay*, and *routing overhead*, are evaluated under various traffic load, and various mobility rate. We compare the performance of the proposed hybrid routing protocol (HARP+TM) with its 95% confidence interval (CI for HARP+TM) with DSR, AODV and OLSR. In the previous chapter, we observe that the TM failed to fully provide load balancing. We solve this problem by giving the priority to the buffer level in the low mobility rate, and to the stability level in the high mobility rate. Note that, in our protocol each node is capable of determining its mobility rate based on the average stability level. In order to show to what extent the performance of HARP with the tuned topology management and with the query localization technique has been improved, we include the results from the previous chapter obtained from the protocol based on the quality of connectivity (marked Routing+TM). In addition to the results of our own simulations for AODV (marked AODV-2 in the figures), we also include a set of results obtained from performance studies done on AODV (marked AODV-1) by the developers of this protocol [70]. This has been done to ensure a fair comparison.

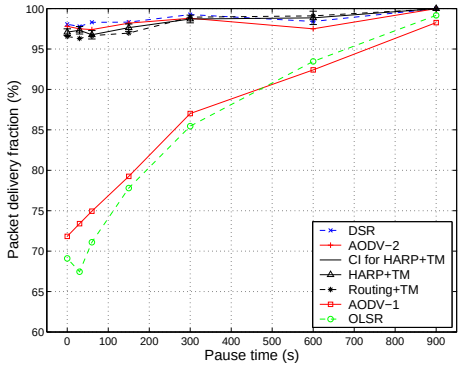
C.1 Packet Delivery Fraction

Packet delivery fraction is defined as the ratio of the data packets delivered to the destination to those generated by the CBR sources. Packet delivery fraction is a very important metric since it describes the loss rate that will be seen by the transport protocols, which in turn affects the maximum throughput of the network. The packet delivery fraction for 10, 20 and 30 sources is shown in Fig. IV-5. For 10 sources, we observe that DSR, HARP+TM and AODV-2 exhibit similar trends. The packet delivery fraction is very high in this case because of the low number of sources in the network which in turn causes a low probability of congestion. As nodes become more mobile, the probability of link failure increases and hence, the number of packet drops also increases. However, due to the load balancing effect triggered by node mobility, the obtained performance of our protocol remains significantly high compared to other protocols. This is not the case for AODV-1 and OLSR, where we notice a drastic reduction in the packet delivery fraction as mobility increases. As the number of sessions is increased, the differences between the protocols start becoming evident. Note that HARP is outperformed only by DSR for the medium and high load scenarios. This is due to the fact that DSR employs route caches, allowing nodes to use backup routes in case of routing failure, which in turn avoid route discovery. HARP maintains a high packet delivery fraction compared to AODV and OLSR. This indicates the robust nature of the protocol and its ability to adapt itself to increasing load. Both AODV and OLSR use minimum hop count as metric. This results in an inherent bias towards the same class of routes causing to congestion. In contrast, the hybrid nature of our protocol as well as the dynamics of the forest together create a load balancing effect resulting in better performance. It is important to note that HARP does indeed show the same packet delivery fraction as DSR for higher loads and high mobility. The effectiveness

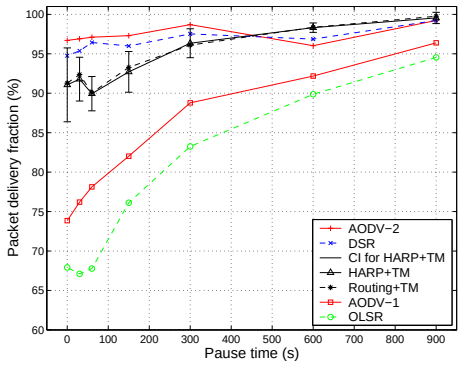
of DSR's aggressive caching strategy is reduced in such scenarios and the advantages of the dynamic nature of forests in HARP are more pronounced. To sum up, when the mobility rate is low, HARP performs as good as any other protocol under all loading conditions. When mobility is high, HARP has almost the same packet delivery fraction as most of the other protocols under investigation. It is worth noting here that this level of packet delivery fraction is achieved with much less delay compared to other protocols (Fig. IV-6).

C.2 Average end-to-end delay of data packets

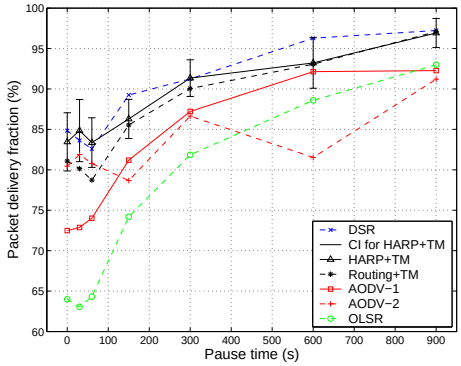
The average end-to-end delay includes all possible delays caused by buffering during route discovery latency, queuing at the interface queue, retransmission delays at the MAC, propagation and transfer times. Generally, there are three factors affecting end-to-end delay of a packet:(i) route discovery time, which causes packets waiting in the queue before a route is found, (ii) congestion state of the network, which causes packets waiting in the queue before they can be sent, (iii) path length, the more number of hops a packet has to go through, the more time it takes to reach its destination. Fig. IV-6 depicts the variation of the average end-to-end delay as a function of pause time. As it can be seen, the general trend of all curves is an increase in delay with high mobility rate. This is mainly due to the increased probability of link failure. In all the three cases, however, note the significantly lower delay incurred in HARP compared with other protocols. When the network load is low and node mobility is high, only the AODV-2 delay is less than HARP. For the medium and high load situations, it is interesting to note that HARP has a very low delay even when the network is static. Generally, in static conditions, delay is pretty high because once routes are set up initially, not many changes occur and hence a high probability of congestion in the network. The forest dynamics of HARP introduce virtual mobility and hence load is redistributed even when nodes remain stationary. For the high mobility case, the HARP outperforms the other protocols for all mobility ranges. Once again, it is the dynamic nature of forests and the hybrid routing strategy that accounts for this behavior. The ability of HARP to overcome the effects of network congestion is a major reason for the lower delay. To summarize, the delay increases with load for all protocols but its important to note that HARP still shows significantly lower delay as compared to others. For the low mobility rate, the difference is more prominent, especially when the network load level is low. As the load increases, other protocols show significant rise in delay whereas in case of HARP, the change is not that much. However, OLSR performs slightly better than HARP but the difference is not much. As a result, the good average delay of HARP is due to its forest-based topology management scheme which prevents congestion in the network by load redistribution.



(a) 10 sources

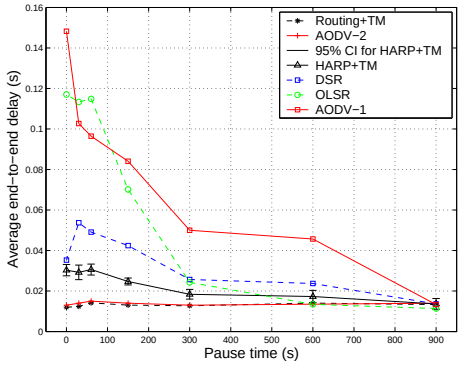


(b) 20 sources

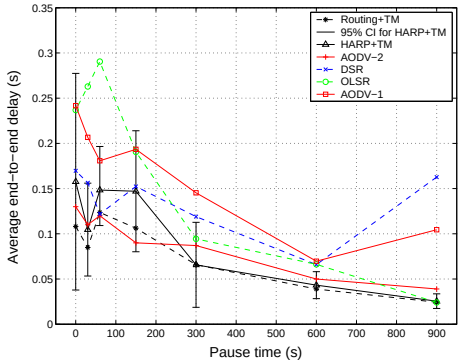


(c) 30 sources

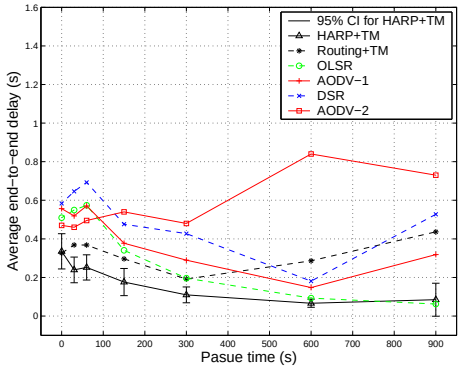
Fig. IV-5. Packet delivery fraction for the 50 nodes with various number of sources



(a) 10 sources



(b) 20 sources



(c) 30 sources

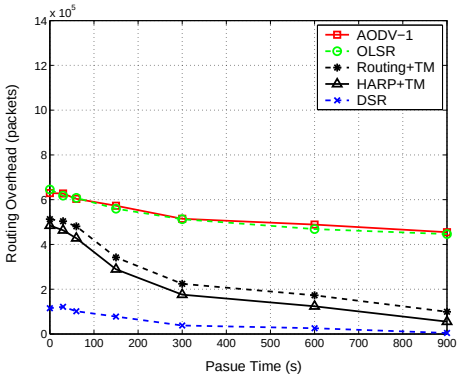
Fig. IV-6. Average data packet delay for the 50 nodes with various number of sources

C.3 Routing overhead

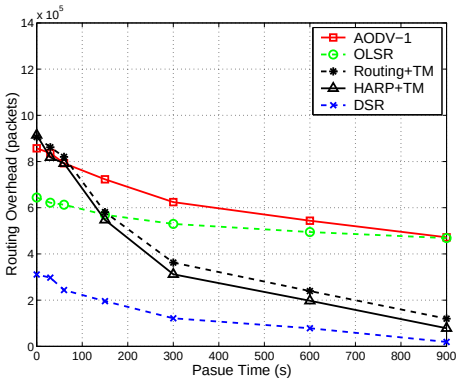
Routing overhead is measured as total number of bytes and packets used for routing process during the simulation. Only overhead stemming from the IP layer is included. Routing Overhead is an important metric to compare these protocols as well, since it measures the scalability of a protocol, the degree to which it will function in a congested or a low-bandwidth environment, and it has a direct impact on network utilization efficiency. Protocols that send large numbers of routing packets and/or bytes can also increase the probability of packet collisions and may delay data packets in network interface transmission queues. Fig. IV-7 & IV-8 illustrate the routing overhead incurred by different routing protocols under different mobility and load conditions. Although DSR has the minimum routing overhead in terms of total number of transmitted packets, most of the times HARP+TM has lower overhead compared to OLSR and AODV. In HARP+TM, a large amount of the packets are due to beaconing process used for the topology management, which are broadcasted locally (see the results in Chapter III). This indicates the reactiveness of HARP+TM to link changes induced by mobility which results in an increased number of routing control packets. Similarly, AODV and DSR detect and react to more link failure when mobility increases However the difference is that, AODV and DSR trigger new route discoveries, while HARP+TM generates more beacons. OLSR always has the same overhead because it does not adapt to increase mobility; the update intervals remain constant. The use of relative distance estimation (RDE) in HARP is one of the main factors reducing protocol overhead. Since RDE limits the scope of route discovery, it helps preventing unnecessary route requests to propagate in the network. When we look at the overhead in bytes, the lowest overhead is incurred by OLSR followed by HARP. Once again, this is due to the more reactiveness of HARP to link changes than OLSR. In DSR, overhead in packets is lower. This is due to the use of route caches to store multiple routes for redundancy. But in DSR, control packets carry full routes and hence packet size is higher. This offsets the advantages of caching and leads to a higher byte overhead.

C.4 Discussion

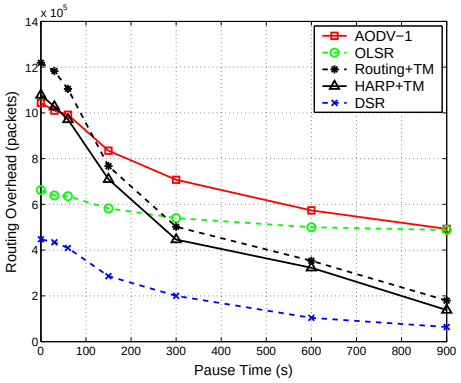
Fig. IV-9 demonstrates explicitly the effect of network load under different mobility rates. It shows that as the traffic load and mobility rate increase in the network, the performance of all protocols decreases. Indeed, the higher traffic load produces a higher collision probability in the network. The situation becomes worse if traffics cross each other. The higher mobility rate, on the other hand, increases the rate of link changes (i.e. both link formation and destruction), which in turn shortens the link stability (or duration). From another point of view, mobility reduces the probability of collision as the traffic load increases. As a result, the network experiences high collision probability in low mobility rate with high traffic load (Fig. IV-9(a) & IV-9(b)). However, the effect of link changes on routing performance is more significant than collision (Fig. IV-9(c) & IV-9(d)).



(a) 10 sources

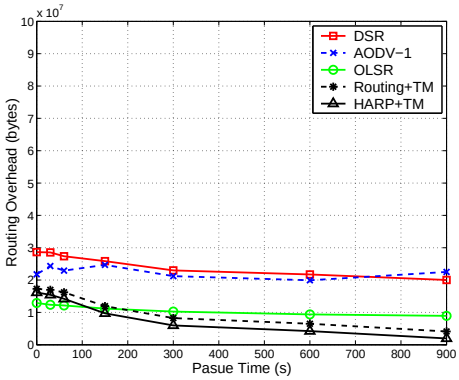


(b) 20 sources

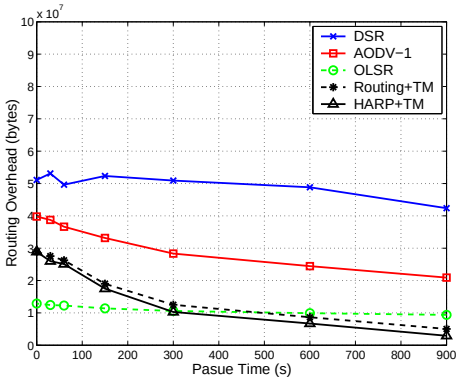


(c) 30 sources

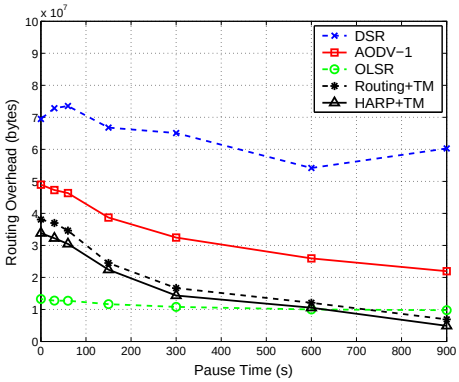
Fig. IV-7. Routing overhead in terms of total number of transmitted packets for the 50 nodes with various number of sources



(a) 10 sources



(b) 20 sources

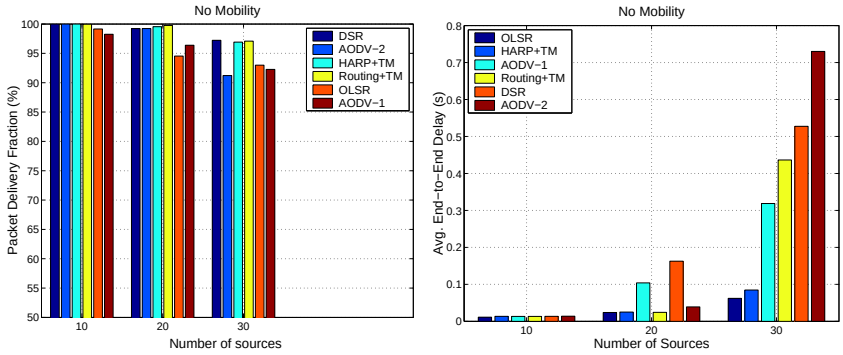


(c) 30 sources

Fig. IV-8. Routing overhead in terms of total transmitted bytes for the 50 nodes with various number of sources

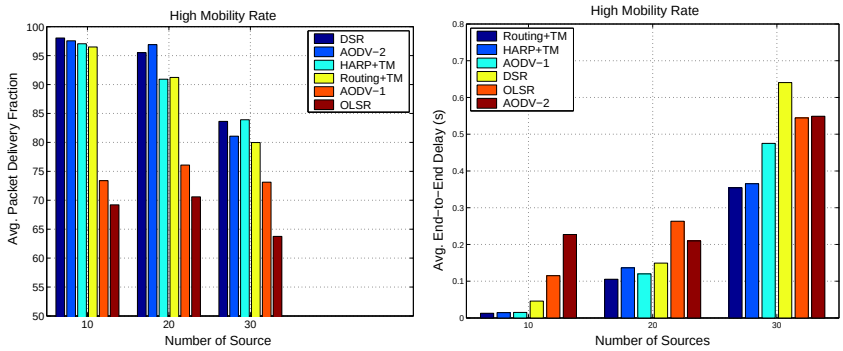
One interesting observation is that for the most protocols the packet delivery fraction uniformly decreases from low mobility rate to medium mobility rate (i.e. from pause time 900 seconds to 150) and then *fluctuates* as the mobility rate increases. This is because the occurrence of *multiuser diversity via relaying* increases with mobility as nodes are likely to be present ubiquitously. This multiuser diversity is best motivated by the information theoretic results of Knopp and Humblet [79]. They showed that the optimal strategy is to schedule at any one time only the user with the best channel to transmit to the base station. Grossglauer and Tse exploit multiuser diversity through relaying for mobile ad hoc networks [8]. The basic idea is for a source to distribute packets to as many different nodes as possible; these nodes relay the packet to the final destination whenever they get close to the destination. Indeed, by using all the other node excluding the source and destination node as relays, the communication is then performed through *two* multiuser links: a downlink from the source to all the relays, and an uplink from the relays to the destination. Note that, the direct point-to-point link is a statistically poor channel, because it is only strong for a small fraction of time when the source-destination pair are close by. Therefore, due to a multiuser diversity effect, the throughput of the downlink is high because at any time, there is likely to be a relay node close to the source to whom the source can transmit the information. The same rule holds for the uplink in relation with the destination. Therefore, the expected path length remains constant. It has to be mentioned that the multiuser diversity via relaying require the information about the neighboring nodes and its results depends strictly on the movement model. The point where the fluctuation begins, roughly after 150 pause time second, depends on network parameters and models, which affect the absolute performance of the protocols in a non-uniform way. By network parameters, we mean: mobility rate, density, size, traffic load; and by network models: traffic pattern, and mobility model.

Gupta and Kumar demonstrated that for a n fixed nodes with random traffic pattern forming a wireless network, the per-node capacity is $\Theta(1/\sqrt{n \log n})$ [7]. Jinyang Li et al. examined the effect of different traffic patterns on per node capacity. They showed that random choice of destination causes a tendency for more packets to be routed through the center of the network than along the edges, which limits the capacity of the network. The authors concluded that the less local the traffic pattern is, the faster per node capacity degrades with the size of network. Grossglauser and Tse studied the effect of mobility on the capacity of ad hoc networks, and showed that capacity improvement can be achieved in the presence of mobility [8]. Camp et al. have provided valuable simulation results demonstrating that the performance of an ad hoc network protocol can vary significantly with different mobility models [9]. They also observed that the choice of mobility model may require a specific traffic pattern which significantly influences the protocol performance. D. D. Perkins et al. evaluated the impact of mobility rate, network size, and traffic load on the protocol performance [10]. They observed that the *number* of traffic sources has the strongest impact on the protocol performance followed by the mobility rate and network size. Also, they noticed that increasing the *rate* of traffic load and the number of traffic source may not result in the same performance results.



(a) Packet delivery fraction for no mobility

(b) Average end-to-end data packet delay for no mobility



(c) Packet delivery fraction for high mobility rate

(d) Average end-to-end data packet delay for high mobility rate

Fig. IV-9. The effect of traffic load on routing performance for high and no mobility rate

We conclude that as the network parameters, mobility rate, density, size, and traffic load (number and rate), increase, traffic patterns should remain local. This implies an upper bound to the path length, which also should remain constant as the network size grows. To keep the traffic pattern local, an appropriate mobility model has to be used so as to maximizes the multiuser diversity.

C.5 Related Work

The following cited research efforts provide the most related performance comparison of routing protocols to our work, as they use the same *ns2*-based simulation environment.

Broch et al. evaluated four ad hoc routing protocols including AODV, DSDV, DSR and TORA [17]. They used 50 node models with similar mobility and traffic scenarios as those used in our simulations. They compared both high movement speed (20m/s) and low speed (1m/s). Traffic loads are kept low (4 packets/sec, $10 - 30$ sources, 64 byte packets). Packet delivery fraction, number of routing packets and distribution of path lengths were used as performance metrics. The results showed that each protocol performs well in some cases yet has certain drawbacks in others. In their simulation, the performance of DSR was very good at all mobility rates and movement speeds, although its use of source routing increases the number of routing overhead bytes required by the protocol. AODV also performs almost as well as DSR at all mobility rates and eliminates source routing overhead, but it becomes more expensive than DSR as the frequency of topology changes increases.

Johansson et al. extended the above work by introducing a new mobility metric that measures mobility in terms of relative rather than absolute speed of a node and pause times [71]. Again, 50 nodes were used. Traffic loads are kept low: only 15 CBR sources. The packet size was 64 bytes and the packet rate was initially 5 packets/second, and then ranged from 5 to 20. Throughput, delay and routing load (in both numbers of packets and bytes) were measured. The overall observation showed that DSR is more effective in low loads, while AODV is more effective at higher loads. The packet-wise routing load of DSR was almost always significantly lower than AODV, however, the byte-wise routing load was often comparable. The authors attributed the comparative poor performance of DSR to the source-routing overheads in data packets. They used small data packets (64 bytes) thus making things somewhat unfavorable for DSR.

Perkins et al. also compared the performance of AODV with DSR in [70]. Two field configuration were used, $1500m \times 300m$ field with 50 nodes and $2200m \times 600m$ field with 100 nodes. They used 512 bytes per packet to make it favorable for DSR. To show the difference between these two protocols in high load, they also compared 40 sources. Packet delivery fraction, average end-to-end delay of data packets and normalized routing load were evaluated. The general observation from their simulation is that for application oriented metrics such as delay and throughput, DSR outperforms AODV in less *stressful* situations, i.e., smaller number of nodes and lower load and/or

mobility. AODV, however, outperforms DSR in more stressful situations, with widening performance gaps with increasing stress (e.g., more load, higher mobility).

Jacquet et al. evaluated the performance of OLSR with AODV and DSR [80]. They considered 50 nodes in 1000×1000 field; 250 simulation time; with the speed of $1-5$ m/s with $0-5$s pause time; 25 CBR sources with the rate of 10 packets/second and size of 64 bytes/packet for the duration of 10 seconds. They also studied the performance of TCP traffic in Manet. Comparison have been done on packet delivery rate, control traffic overhead, route length, and performance under TCP traffic. They demonstrated that the performance of OLSR is better than that of AODV and roughly similar to DSR for the low mobility rate, while AODV and DSR outperform OLSR in high mobility rate. They observed that proactive protocols outperform reactive protocols in heavy traffic load. For the TCP traffic, they also observed that OLSR as a proactive protocol performs significantly better than both AODV and DSR.

D CONCLUSION

We have proposed a hybrid routing protocol for mobile wireless ad hoc networks, which benefits from the separation of topology management and route determination. We adopt a modular approach in order to address the competing design goals of minimum delay, minimum overhead and maximum packet delivery fraction. HARP efficiently combines two routing strategies: proactive and reactive depending on whether the destination node belong to the same zone as the forwarding node. The forest and localization properties of HARP makes the protocol fully bandwidth efficient because control messages do not propagate throughout the network. Therefore, HARP remains scalable as the density and number of nodes in the network increases.

We have compared the performance of this hybrid protocol with AODV, DSR and OLSR. The key metrics for evaluation are packet delivery fraction, end-to-end delay and routing overhead. The obtained results show that our hybrid protocol outperforms other protocols. The packet delivery fraction for HARP is close to the same level as the best-performing DSR and much better than AODV and OLSR. Furthermore, this level is achieved without any degradation in the other metrics. Dynamic topology management in HARP is a key factor as it has a load balancing effect. The end-to-end delay in the case of HARP is the best among all protocols under consideration. Moreover, the the difference of the end-to-end delay obtained by HARP with other protocols is very high when mobility is low. This can be attributed to the effect of the topology management mechanism in HARP. Although the overhead of HARP in terms of total number of transmitted packet is higher than that of DSR and OLSR in more stressful situations, most of the packets are due to beaconing process used for topology management. DSR reduces the number of route discovery cycles by using caching mechanism. OLSR keeps the rate of the topology update constant which in turn decreases the reactiveness of the protocol to link changes induced by mobility. When we look at the overhead in bytes, the lowest overhead is incurred by OLSR following by HARP. This is because

OLSR does not adapt to increase mobility, while HARP detects and reacts to more link failure when mobility increases. The drawback of DSR is that the routing packets carry the full route (source routing). This is reflected in the high byte overhead incurred by DSR. In this case, HARP is able to keep a low overhead in bytes while still minimizing the delay. Overall, we can safely say that our hybrid protocol performs very well under different load and mobility conditions.

We will continue to explore the performance of HARP under new network parameters, models, and performance metrics. One of the issues that we would like to see is the behavior of HARP in the presence of symmetric links. We hope that our results encourage further research and development for hybrid routing protocols and HARP specially.

CHAPTER V

Quality of Service Support

Contents

A Background on Quality-of-Service Models 87
B Proposed Quality-of-Service Model 88
B.1 Assumption 88
B.2 Intuition 89
B.3 Network Layer Metrics 90
B.4 Quality of Service Classes and Metrics' Mapping 91
B.5 Application Metrics 92
C Performance Evaluation 93
D Additional Mechanisms 95
D.1 Path Stability Period 95
D.2 Service Differentiation in Nodes 95
E Extensions for Routing Protocols 97
E.1 Proactive Approach 97
E.2 Reactive Approach 98
F Conclusion and Future Work 99

Abstract—*A lightweight efficient quality of service model is suggested for mobile ad hoc networks. It generates paths with respect to the network quality to avoid unbalanced network utilization while minimizing the resource consumption, and then selects according to the application requirements. In this model, network metrics and application metrics are used as the additional constraints to the conventional ones to determine paths between a source and a destination, because we believe that the quality of service that an application requires depends on the "quality" of the network. Due to link failure caused by lack of ad hoc network resources and node mobility on a path, this quality should not only reflect the available resources on a path but also the stability of that path. The simulation results show that the proposed QoS model improves significantly the routing performance, specially in terms of delay, over the standard shortest path model.*

Keywords—*Mobile ad hoc networks, quality of service, quality of connectivity, network metrics, application metrics, connectivity.*

WITH the introduction of real-time audio and video applications, specifically two-way voice communications (e.g. telephony) into mobile ad hoc networks, the communication path that is selected between the nodes has to meet additional constraints (e.g. delay). In addition to the destination node, the application must also supply the constraint parameters (i.e. its QoS parameters) to the routing protocol so that a suitable path can be found. The routing protocols that support QoS must be *adaptive* to cope with the time-varying topology and time-varying network resources. For instance, it is possible that a route that was earlier found to meet certain QoS requirements no longer does so due to the dynamic nature of the topology. In such a case, it is important that the network intelligently adapts the session to its new and changed conditions.

According to RFC 2386 [81], quality of service means providing a set of service requirements to the flows while routing them through the network. This definition may not be valid in mobile ad hoc networks since even the Internet today, with high-speed high-quality fixed communication links, is unable to deliver guaranteed end-to-end rates or delay [82]. For mobile ad hoc networks, with time-varying low-capacity resources, the notion of being able to guarantee these form of QoS is not plausible. Instead, applications must adapt to time-varying low-capacity resources offered by the network. Therefore, the quality of service that an application requires depends on the *quality* of the network. Hence, the quality of service in mobile ad hoc network can be redefined as *a set of parameters in order to adapt the applications to the quality of the network while routing them through the network.* Therefore, quality of service routing is a routing mechanism under which paths are generated based on some knowledge of the quality of the network, and then selected according to the quality of service requirements of flows. Hence, the task of QoS routing is to optimize the network resource utilization while satisfying application requirements.

Because the quality of service that an application requires depends strictly on the *quality* of the network, we define network layer metrics (NLM) and application layer metrics (ALM) as the additional constraints to the conventional ones to determine paths between a source and a destination [57, 58]. Network metrics are used to determine the

quality of a path to route the data traffic using the quality of individual node in that path. We use concave and additive functions to represent the network metrics corresponding to a path given the values of these metrics for individual nodes on that path. We define three ALM, in addition to the standard best-effort, to meet specific application requirements. In order to be able to compute these metrics, a reasonable combination of network metrics is mapped onto the application metrics. Once the quality of paths is determined by the network metrics, the application metrics select exactly one path in order to meet the desired requirements. In this sense, the application metrics become secondary metrics and support applications.

In the following, some background information about the basic QoS model is presented. The intuition of the proposed QoS model is presented followed by its detailed description. Various simulation results are presented. The extension of the model for proactive and reactive schemes is described. Finally, some ideas for future work along with the conclusion are provided.

A BACKGROUND ON QUALITY-OF-SERVICE MODELS

The presence of mobility implies that links make and break often and in an indeterministic fashion. This dynamic nature makes routing and consequently QoS support in these networks a challenging task [83, 84, 85, 86]. Further, since the *quality* of mobile nodes (in terms of their connectivity to the network, e.g. enough buffer, battery) varies with time, present QoS models for wired networks are insufficient for such networks [87]. *Integrated services* (IntServ) [88] and *Differentiated services* (DiffServ) [89] are the two basic architectures proposed to deliver QoS guarantees in the Internet. A variant of these two architectures: a Flexible QoS Model for Manet (FQMM) [90] has been proposed for ad hoc networks. Below is a short summary of IntServ, DiffServ, and FQMM.

- ***Integrated Services***—IntServ architecture allows sources to communicate their QoS requirements to routers and destinations on the data path by means of a signaling protocol such as RSVP [91, 92]. Hence, IntServ provides per-flow end-to-end QoS guarantees. IntServ defines two service classes: *guaranteed service* [93] and *controlled load* [94], in addition to the *best effort* service. The guaranteed service class guarantees to provide a maximum end-to-end delay, and is intended for applications with strict delay requirements. Controlled load, on the other hand, guarantees to provide a level of service equivalent to best effort service in a lightly loaded network, regardless of network load. This service is designed for adaptive real-time applications. As is the case in the Internet, IntServ is not appropriate for mobile ad hoc networks, because the amount of state information increases proportionally with the number of flows, which results in scalability problems.

- *Differentiated services* – DiffServ architecture avoids the problem of scalability by defining a small number of per-hop behaviors (PHBs) at the network edge routers and associating a different DiffServ Code Point (DSCP) in the IP header of packets belonging to each class of PHBs. Core routers use DSCP to differentiate between different QoS classes on per-hop basis. Thus, DiffServ is scalable but it does not guarantee services on end-to-end basis. This is a drawback that hinders DiffServ deployment in the Internet, and remains to be a drawback for Manet as well, since end-to-end guarantees are also required in Manet. In DiffServ, we can identify three different classes: *expedited forwarding*, *assured forwarding*, and *best effort*. Expedited forwarding provides a low delay, low loss rate, and an assured bandwidth. Assured forwarding provides guaranteed/expected throughput for applications, and best effort which provides no guarantee.

DiffServ and IntServ require accurate link state (e.g. available bandwidth, packet loss rate , delay, and etc.) and topology information. The time-varying low-capacity resources of the ad hoc networks make maintaining accurate routing information very difficult. However, we think that a quality of service model for Manet should benefit from the concepts and features of the existing models in order to build on a model that satisfy such networks. A variant of these two architectures, called a flexible QoS model for Manet (FQMM), has been proposed for mobile ad hoc networks [90]. FQMM defines three type of nodes: an ingress node which sends date, an interior node which forwards data to other nodes, and an egress node which is a destination. Obviously, each node may have multiple role. This model selectively uses the per-flow state property of IntServ, and the service differentiation of DiffServ. That is to say, for applications with high priority, per-flow QoS guarantees of IntServ are provided. On the other hand, applications with lower priorities are given per-class differentiation of DiffServ. Therefore, FQMM applies a hybrid provisioning where both IntServ and DiffServ scheme are used separately. In FQMM, both IntServ and DiffServ scheme are separately used for different priority classes. Therefore, the drawbacks related to IntServ and DiffServ remain to be a drawback in FQMM.

B PROPOSED QUALITY-OF-SERVICE MODEL

B.1 Assumption

In the proposed quality of service model, a mobile ad hoc network with a symmetric environment where all nodes have similar capabilities is considered. By capabilities, we mean transmission range, battery life, processing capacity, buffer capacity, and speed of movement. However some modifications are needed in order to validate the model in a fully asymmetric environment.

B.2 Intuition

Unlike fixed networks such as the Internet, quality of service support in mobile ad hoc networks depends not only on the the available network resources but also on the node mobility. This is because mobility may result in link failure which in turn may result in a broken path. Furthermore, mobile ad hoc networks potentially have less resources than fixed networks. Therefore, more criterion are required in order to capture the quality of the links between nodes.

We suggest a quality of service model that separates the network layer metrics from the application layer metrics because we believe that the quality of service that an application requires depends on the *quality* of the network. Our model extends the core routing functionalities, mainly path generation and path selection procedures (c.f. Chapter IV), for supporting quality of service. In the path generation process, the quality of a path to route the data traffic is computed using the quality of each individual node in that path. This quality is measured by the network metrics. Three network metrics are defined: hop count, path buffer, and path stability. Hop count corresponds to the number of hops required for a packet to reach its destination. The hop count of the path is very important because the more hops a packet traverses, the more resources it consumes. For example, a 1-Mbps flow that traverses two hops consumes twice as many resources as one that traverses a single hop. The two other metrics: path buffer and path stability, determine the buffer level and stability level used to described the quality of connectivity for a path instead of for a node (c.f. Chapter III). So, the network metrics are computed during the path generation procedure. Indeed, we use concave and additive functions to represent the network metrics corresponding to a path given the values of these metrics for individual nodes on that path. With the knowledge of the quality of paths, application metrics select the most suitable path according to the desired QoS class. For this purpose, application requirements are classified into three QoS classes: I, II, & III with a descending priority. Then, each class is mapped onto appropriate metrics at the application layer including delay, throughput, and enhanced best-effort; respectively. In order to be able to compute these metrics, a reasonable combination of network metrics is mapped onto the application metrics. In fact, the path buffer and the hop count are used to compute a path for the class I and II, while class III uses the path stability and hop count to determine a path. Fig. V-1 shows the mapping between QoS classes (I, II, & III); delay, throughput, and enhanced best-effort at the application layer; hop count, buffer level, stability level at the network layer.

In order to keep the routing overhead low and support fast routing decisions in QoS routing, a *state* is associated to the available network resources. In the path generation phase, the nodes use this state information to generate paths according to the available network resources. Then in the path selection phase, this state is used in conjunction with the desired QoS class to select the most suitable path according to the application requirements.

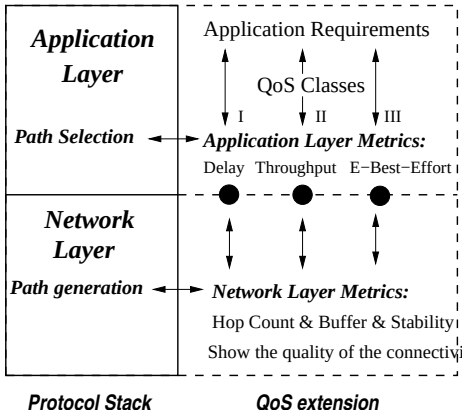


Fig. V-1. Global view of layered quality of service model

B.3 Network Layer Metrics

Network metrics characterize the quality of paths in terms of their available resources and the stability of such resources. This is because the quality of service that an application requires depends strictly on the *quality* of the paths. Three network metrics are defined: hop count, path buffer, and path stability; which are computed during the path generation procedure in order to indicate the kind of service requirements that can be met by the generated paths. Hop count defines the hop-wise distance between source and destination, and it used to minimize the network resource utilization. Path buffer and path stability extend the notion of buffer level and stability level used to define the quality of connectivity for a path (c.f. Chapter III). They are used in conjunction with the hop count metric so as to avoid unbalanced network utilization. Hence, the network metrics provide a trade-off between load balancing and resource conservation.

In the path generation phase, network metrics are propagated through the nodes of generated paths. With the exception of hop count- which is simply the number of hops required to reach the destination, concave and additive functions are used to represent the network metrics corresponding to a path given the values of these metrics for individual nodes on that path. Suppose P is a path between source node s and destination node d (that is P is a sequence of (non-repeating) nodes, hence $P =< s, n_1, \ldots, n_k, d >$). One way to estimate the value of these metrics for P is:

$$P.hop = \sum_{n \in P \setminus \{s\}} 1 \tag{V-1}$$

$$P.buffer = \min_{n \in P \setminus \{s\}} n.buffer \tag{V-2}$$

$$P.stability = \sum_{n \in P \setminus \{s\}} \frac{1}{n.stability} \tag{V-3}$$

The stability level of P makes a trade-off between the most stable path when network mobility is high and shortest path when network mobility is low. The buffer level of P is represented by the node with the least buffer on P. Indeed, this is appropriate for the route generation procedure, since a route is rendered broken even if one intermediate node has no buffer.

Recall from Chapter III, that a two-bit code is used to capture the buffer and stability levels and classify them into *high*, *medium*, *low* and *selfi sh* states respectively. In computing the corresponding codes for these metrics on paths, we first map these states to the set $\{3, 2, 1, 0\}$ respectively, evaluate the metrics for the paths as given above, and then unmap the result back to these codes using a ceiling function.

B.4 Quality of Service Classes and Metrics' Mapping

In order to incorporate application requirements in the path selection procedure, we have to translate them into meaningful metrics. In our model, initially applications are classified into different QoS classes. Then, each class is mapped onto the appropriate application metrics that specify the application QoS constraints. Afterward, a reasonable combination of network metrics is mapped onto each QoS classes and therefore to its ALMs. This enables ALMs to select exactly one path based on the desired QoS class which is more likely to meet application requirements.

We define three QoS classes for the destination to select the best available path [95]. Class I corresponds to applications that have strong *delay* constraints, for example applications with real-time traffic such as voice. The corresponding service of this class in DiffServ is referred to as expedited forwarding and in IntServ to as guaranteed service. The well-known port for this class is VAT. We map this class to the delay metric at the ALMs, and to the buffer level and hop count at the NLMs. Therefore, the path selection procedure attempts to extract a path that has minimum delay on the basis of the average buffer level and hop count. We assume that queuing delay of a packet is a good estimate of its end-to-end delay. Class II is suitable for applications requiring high *throughput* such as video or transaction-processing applications. The service of this class is referred to assured forwarding in DiffServ and controlled load in IntServ. FTP and HTTP are the well-known ports for this class. We map this class to the throughput metric at the ALMs, and to the buffer level and hop count at the NLMs as in the first class. Finally, Class III has no specific constraints. These classes

are referred to the *best-effort* in both architectures. We map this class to the *enhanced best-effort* at the ALMs, and to the stability level and hop count at the NLMs. The rationale behind the enhanced best-effort service is that it compromises between the most stable path and the shortest path. In fact, it selects the most stable path when the network mobility is high and the shortest path when the network mobility is low. Table V-1 shows the defined QoS classes together with their ALMs constraints and the corresponding NLMs. A detailed description of how the application metrics are mapped on to the network metrics is given in the following section.

TABLE V-1. QOS CLASSES & MAPPING

Priority Classes	ALMs	NLMs
Class I	Delay	Buffer & Hop Count
Class II	Throughput	Buffer & Hop Count
Class III	Enhanced Best-Effort	Stability & Hop Count

B.5 Application Metrics

The application metrics are employed by the path selection procedure. Indeed, they give a *reflection* of paths' class for the application based on the information provided by the network metrics. This reflection is a basis to compare the generated paths, and select the most suitable one.

- ***Delay***—the delay is the total latency experienced by a packet to traverse the network from the source to the destination. At the network layer, the end-to-end packet latency is the sum of processing delay, packetization, transmission delay, queuing delay and propagation delay. Queuing delay contributes most significantly to the total latency and all other delays are negligible. Hence, it is appropriate to estimate the total latency experienced by a packet by the queuing delay experienced by the packet as it moves from the source to the destination. From the two metrics: path buffer (say b) and hop count (say h) computed using equation V-1 and V-1, one can estimate the queuing delay experienced by a packet as in equation V-4, where r is the buffer size and c represents the total link throughput. Recall that b represents the unallocated buffer, and hence $r - b$ denotes the buffer occupancy.

$$P.delay = h \cdot (r - b)/c \tag{V-4}$$

$$P = \min_{for\ all\ P} P.delay \tag{V-5}$$

- ***Throughput***—the throughput is defined as the rate at which packets are transmitted in the network. It can be expressed as the peak rate or the average rate. Note

that the throughput is reduced because of packet loss, that may be caused by link failure due to node mobility and lack of resources. Assuming that each node's throughput is c, the throughput for an end-to-end connection can be estimated as:

$$P.throughput = \frac{c}{2h \cdot (r - b)} \tag{V-6}$$

$$P = \max_{for\ all\ P} P.throughput \tag{V-7}$$

- ***Enhanced Best-Effort***—the enhanced best-effort service provides no service guarantees for the applications. It compromises between the most stable path in high mobility case and shortest path in low mobility case. In the former case, it establishes the most stable path so as to improve delay performance caused by link failure caused due to the node mobility. In the latter case, it selects the shortest path to minimize network resource utilization since the more hops a flow traverses, the more resources it consumes. We evaluate this metric in the following equation, where s represents the path stability.

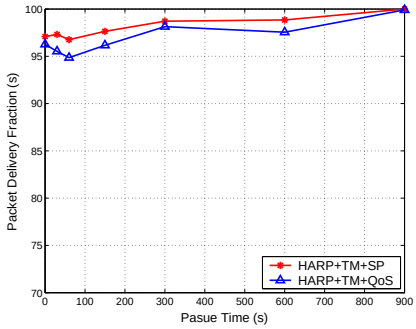
$$P.enhanced\_best\_effort = s \tag{V-8}$$

$$P = \min_{for\ all\ P} P.enhanced\_best\_effort \tag{V-9}$$

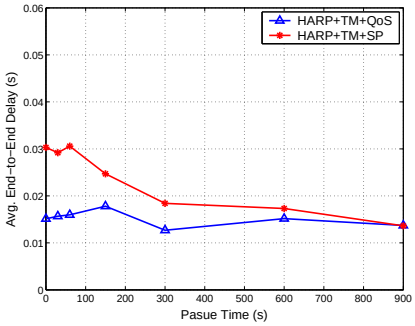
C PERFORMANCE EVALUATION

We use a simulation model based on *ns-2* in our performance evaluation [67]. In the following, we compare the performance of the QoS routing (marked HARP+TM+QoS) with the shortest path routing (marked *HARP+TM+SP*). In *HARP+TM+QoS*, each source randomly selects a metric among delay, throughput, stability, and shortest path when it initiates a path discovery procedure; while *HARP+TM+SP* always chooses the shortest path. Two key performance metrics are evaluated: packet delivery fraction, and average end-to-end packet delay, under various mobility rate and traffic load. We used the same configuration as in the previous simulations: 50 nodes in 1500×300; 900 simulation time; with the speed of $0 - 20$ m/s with 7 pause times; $10, 20$ and 30 CBR sources with the rate of 4 packets/seconds and size of 512 bytes for the whole simulation time.

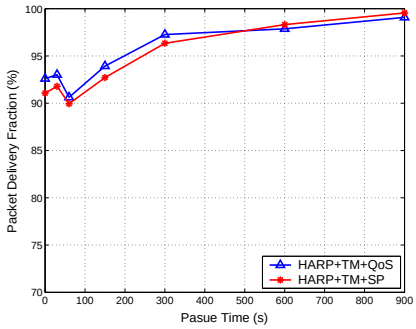
Fig. V-2 shows the packet delivery fraction and the average packet delay under different traffic load and mobility rate. Under light traffic load, packet delivery fraction of the SP metric outperforms the QoS (Fig. V-2(a)). This is because the QoS render a longer path than SP which in turn increases the chance of link failures as nodes become more mobile. As a result, the number of packet drops in QoS is higher than SP. Remember from the previous chapters that the path discovery is only propagated from zone to



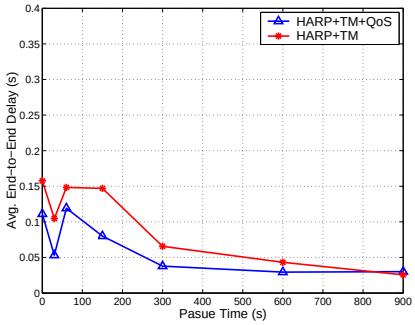
(a) 10 sources



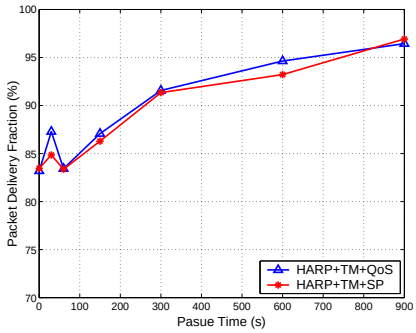
(b) 10 sources



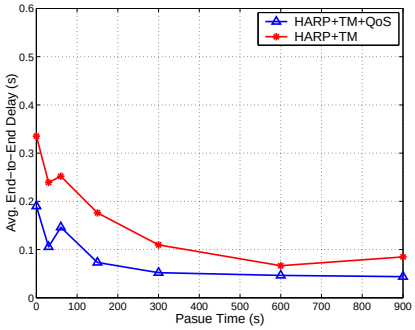
(c) 20 sources



(d) 20 sources



(e) 30 sources



(f) 30 sources

Fig. V-2. Packet delivery fraction for the 50 nodes with various number of sources

zone, and the average path length within a zone is no longer than 2 times of the optimal path length. However, the advantage of QoS becomes apparent when the traffic load gets heavy. With the SP, the tendency for more packets to be routed through the center of the network than along the edges increases as the traffic load increases. As the network traffic becomes heavy, these routes becomes congested, causing packets to be delayed or dropped. On the other hand, *HARP+TM+QoS* tries to distribute the traffic and take the advantage of edge routes. This is why the delay performance of *HARP+TM+QoS* is always better than *HARP+TM+SP* (Fig. V-2(b), V-2(d) & V-2(f)). However, the relative performance of both protocols with respect to packet delivery fraction for medium and high traffic load is almost similar (Fig. V-2(c), & V-2(e)). Therefore, we can conclude that the QoS mainly improves delay performance.

D ADDITIONAL MECHANISMS

D.1 Path Stability Period

The stability level can also be seen as a *widthwise* metric since it is possible to consider this metric for all QoS classes. It can be used to capture the *duration* for which the communication between the source node and destination node may remain unbroken. This duration is called *stability period*. The rationale relies on the fact that a high stability indicates (with a large probability) a low state of node mobility, while a low stability indicates (with a large probability) a high state of mobility. For a path P between the source node s and the destination node d, we estimate the stability period as:

$$SP = P.stability \cdot T \tag{V-10}$$

Where $P.stability$ is estimated using a concave function (refer to equation IV-1), and T is the beaconing period for stability evaluation (see Chapter III H.1). Note that if the stability period of a particular path is equal to the beaconing period T, then this implies that all nodes on this path are stable, and hence the connection is expected to remain unbroken for the entire period T. Stability period is used to estimate the *life* of a selected path. If the stability period of a path is low, then a new path generation phase is expected to be triggered soon, because it is likely that a link on this path would go down (it has low stability). This is a desired behavior for the delay sensitive application. On the other hand, if the stability period of a path is high, then this path has a long *life*.

D.2 Service Differentiation in Nodes

The model differentiates services and provides soft guarantees to network resources for an admitted application by using a *class-based weighted fair queuing* (CB-WFQ)

at intermediate nodes.

In this section, we propose an analogy of DiffServ architecture proposed for the Internet, which extends our model to provide a mechanism that guarantees the network resources for an admitted application on per-class basis.

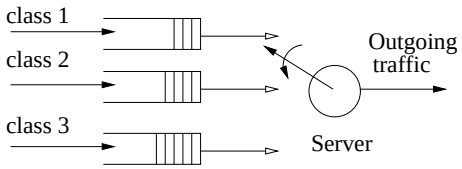


Fig. V-3. Service Differentiation in an Ad hoc Node

To achieve this, we propose using the *class-based weighted fair queuing* (CB-WFQ) scheduling in ad hoc nodes. Class-based WFQ is the extension of the standard weighted fair queuing (WFQ) [96, 97] functionality to provide support for user-defined traffic classes. In CB-WFQ, a queue is reserved for each QoS class, and traffic belonging to a class is directly forwarded to the queue for that class. Fig. V-3 shows service differentiation in a node. After packets are assigned to their corresponding queues, they receive prioritized service based on user-configured weights assigned to the queues. We define QoS classes in Tables V-1. In our approach, classification is performed by a source node. A source node assigns a QoS class to a packet by tagging a (two bit) code to the IP header of each packet belonging to an application. No further classification is required at the intermediate nodes. Upon arrival at an intermediate node, a packet is directly placed to the queue associated to its QoS class. Hence, each queue buffers packets belonging to the same QoS class. In this model, the packets that reside in the same queue may belong to different applications with the same QoS class.

Finally, the server services packets from different queues based on the priority of the queue, which corresponds to the weights set for each queue in every node. Example of weights for each queue at the node can be set such that, class I service occupies 60% of the CPU times, class II service 30%, and class III service gets 10%. The weights in CB-WFQ are necessary to guarantee minimum bandwidth to each QoS class, this also prevents complete starvation of applications with lower priorities. Furthermore, the unused capacity in CB-WFQ is shared amongst other classes proportional to their weights. Traffic belonging to class I has strong delay constraints, and hence must be forwarded with a priority, and this is captured by service differentiation. Hence, using CB-WFQ, a node guarantees QoS resources of an admitted application through scheduling.

E EXTENSIONS FOR ROUTING PROTOCOLS

According to our quality of service model, we re-establish the core routing functionalities described in Chapter II. They consist of:

- ***Path Generation***—(i) including NLMs into routing state information, (ii) generating paths according to NLMs;
- ***Path Selection***— (i)mapping desired QoS class to ALMs, (ii) computing ALMs from NLMs,(iii) selecting the most suitable path according to ALMs;
- ***Data Forwarding***— assigning data to its appropriate queue according to its QoS class.

In view of the steps presented above, the main question is: *How can a routing protocol be extended to support QoS?* For this purpose, we explore some issues in ad hoc routing. One of the issues with routing in ad hoc networks concerns whether nodes should keep track of routes to all possible destinations, or instead keep track of only those destinations that are of immediate interest. A node in an ad hoc network does not need a route to a destination until that destination is to be the recipient of packets sent by the node, either as the actual source of the packet or as an intermediate node along a path from the source to the destination. In the literature related to routing strategies used in mobile ad hoc networks, we find two main classes of routing including *proactive* and *reactive* [2, 15]. For the sake of clarity, we describe briefly each class together with the necessary modification that has to be made to support our QoS model.

E.1 Proactive Approach

In proactive routing protocols, each node continuously keeps track of routes for all destinations in the network. Routing information is periodically *generated* throughout the network in order to maintain routing table consistency. When an application starts, a route can be immediately *selected* from the routing table. Proactive routing methods can be classified as two primary algorithms: *link-state* and *distance-vector* [19, 12]. We describe our extension for each algorithm.

In the link state approach, each node maintains a view of the network topology with a cost for each link. In order to support QoS, each node periodically broadcasts the NLMs of its outgoing links to *all* other nodes in the network. Indeed, NLMs replace the link costs in the classical link-state algorithm. As a result, each node has a labeled graph which represents the topology of the entire network with the QoS state of each node as well as the number of hops for each destination. In order to choose the next hop for a particular destination meeting application requirements, a node maps the link costs as the associated NLMs onto the ALMs according to the desired QoS class. Then, it applies a shortest-path algorithm with the associated ALMs to the labeled

graph to determine the next hop. This next hop is used to *forward* data traffics toward the desired destination. During the forwarding process, the data traffics are assigned to their appropriate queues according to their QoS classes. For example, for a delay sensitive application, in order to determine the next hop, a node estimates the cost of an outgoing link from node u to node v as $v.h \cdot (r - \overline{b})/c$. Note that this cost reflects the queuing delay experienced by the packet at node v if it were routed to its destination along v. Hence, a shortest path with the cost of links defined in such a manner will yield the path with minimum queuing delays.

In distance vector algorithms, each node i maintains for each destination x, a set of distances $d_{ij}(x)$, where j is a neighbor of i. In order to support QoS for distance-vector algorithm, each node broadcasts a set of NLMs instead of a set of distances *only* to its neighborhood. Thus, each node regularly updates the NLMs of its outgoing links. In order to route a packet to destination x, node i picks the neighbor k as the next hop if $d_{ik}(x) = \min\{d_{ij}(x)\}$, as in the classical Distributed Bellman-Ford (DBF) algorithm [14, 98]. However, unlike the present distance vector algorithms, the NLMs are taken into account to determine the next hop. Indeed in the process of next hop selection, each node maps the desired QoS class to ALMs, and then compute ALMs according to NLMs. The succession of next hops chosen in this manner leads to x along the shortest path. To keep the distance estimates up to date, each node monitors the NLMs of its outgoing links and periodically broadcasts its current estimate of the shortest distance from itself to every other node in the network. Once next hop is selected, data forwarding occurs within the appropriate queue.

E.2 Reactive Approach

In reactive routing protocols, a node *generates* all possible paths throughout the network only when necessary in response to data traffic demands at a source node. This process is completed once all possible paths have been examined in order to *select* the most suitable one. Therefore, for every unknown destination all possible paths have to be generated. In order to support QoS for reactive routing, the source nodes computes the NLMs of all possible paths for the destination that it wishes to communicate with. Once the NLMs are computed, a path can be selected based on the desired QoS class thanks to the mapping between QoS classes, ALMs, and NLMs. Note that, the QoS class identifies the service that is required by the applications. It is also used to assign packets to their appropriate queues during the routing process. Finally, source node uses the selected path for data forwarding.

To give the intuition of our model, for example if the destination receives two paths (P_1 and P_2) with different hop counts (h_1 and h_2, $h_1 > h_2$), different buffer levels (b_1 and b_2, $b_1 < b_2$), and equal stability level; then it may select P_2 if the application is FTP (since FTP applications require low loss rate), while it may select P_1 if the application is voice (since audio applications require low latency).

F CONCLUSION AND FUTURE WORK

We have proposed a quality of service model that considers the characteristics of Manet and tries to emulate both end-to-end service management of IntServ while maintaining the scalability and per-class service differentiation of DiffServ. It defines the network metrics from the application metrics as the additional constraints to the conventional ones in order to responds to two different requirements: at the network side, it maximizes network life time by evenly distribute the traffic throughout the network; and at the application side, it selects a path which is more likely to meet application requirements. As a result, our QoS model consider the current quality of the network in addition to application requirement to establish end-to-end communication path.

We have evaluated the performance of our QoS routing protocol with its shortest path version. The simulation results have shown that the gain of QoS in comparison with SP is more significant in terms of end-to-end packet delay than packet delivery fraction. It is worthwhile to evaluate the performance of the QoS model without the topology management strategy. Therefore, the question of whether the model by its own could provide load balancing in the network remains open.

CHAPTER VI

Conclusion

ROUTE determination and topology management are two key issues in designing a routing protocol for the mobile ad hoc networks. Several research efforts have shown the difficulty of coping with these issues and their associated challenges. However to the best of our knowledge, these factors have always been dealt as one single problem, in particular by topology-based routing protocols. Most of the solutions are often biased either on the route determination to improve packet delivery fraction or on the topology management to improve the delay performance. For instance, the effect of mobility is different on the performance of AODV, DSR, and OLSR [70]. In the presence of mobility, link failure can happen frequently. OLSR does not adapt to increased mobility, the update interval remain constant. AODV and DSR on the other hand detect and react to more link failures when mobility increases. In AODV, link failures trigger new route discoveries whose frequency is directly proportional to the number of route breaks. This makes the overhead of AODV strictly dependent to the network conditions. In more stressful situations (i.e. large number of nodes, sources, and/or mobility), increasing this frequency does not improve the protocol performance as this may increase the probability of collision resulting link failures for other traffic. The reaction of DSR is to delay the route discovery procedure until all cached routes fail. But with high mobility, the chance of the caches being stale is quite high results in significant performance degradation. We believe that route determination and topology management have to be tackled independently. Routing protocol must always stay tuned to the network conditions and react accordingly. That is why, an architecture that separates topology management from route determination is proposed. In this architecture, topology management dynamically adjusts the network topology with respect to the quality of network; whereas route determination finds the optimal path *on the*

adjusted topology to meet application requirements. The principle of adjusting network topology is that not all connectivity will be suitable when the network situation becomes stressful. Hence, the route discovery is *only* propagated on the adjusted topology. Thus, mobility may only render a longer path; it increases neither the frequency of route discovery, nor the delay of route acquisition (because the path is not congested).

The proposed architecture tackles the issues of delay and overhead independently from the routing performance issue. The control overhead of a routing protocol is directly related to the the scope of route discovery and the mechanism used for propagating route requests. For example, if flooding is used for sending requests, it will lead the propagation of route requests in an omni-directional way throughout the network. Thus, the request will travel even to those parts of the network that are no way near the destination. This constitutes a wastage of resources. Hence, one way of reducing overhead is by using a mechanism that does not induce flooding. There is another dimension to the flooding issue that is related to the scope of the request. If we are able to localize the discovery process, it will ensure that the requests will not travel too far beyond the distance at which the destination is located. Keeping these considerations in mind, we propose a forest-based topology management scheme that attempts to minimize the effect of flooding. Furthermore, we employ a distance estimation algorithm to limit the scope of the route requests. This can help in reducing overhead but it may have an adverse impact on the delay performance because the estimates may not always be true. In the latter case, further route discovery cycles will be required over a bigger area, resulting in higher delay.This motivates us to use the concept of zones along with a proactive intra-zone routing strategy. The delay performance will be greatly improved by this approach. So far, we have been focusing on the trade-off between control overhead and delay. We also need to maximize the network utilization by maximizing the packet delivery fraction in the presence of frequent link failures. The single biggest reason behind poor packet delivery performance is the variable nature of the wireless channel. The problem is further exacerbated by the dynamic network topology. We use the notion of quality of connectivity to address this issue. This metric can be used to indicate the quality of nodes on the basis of a set of suitable criteria. The idea is to use the metric to select highest quality links that are more stable and have a better chance of ensuring high packet delivery performance. Furthermore, an adaptive approach is used because quality may change over time. This implies that the resulting forest structure will be dynamic which in turn, has a load balancing effect on the network.

This study have shown that the proposed architecture and protocols outperform AODV, DSR, and OLSR. Delay performance is improved up to 30 percent in the shortest path routing and up to 70 percent in the QoS routing in comparison with the reactive and proactive protocols under study. This improvement level is achieved without any degradation in the other performance metrics. The packet delivery fraction for HARP is close to the same level as the best-performing DSR and much better than AODV and OLSR. Although the overhead of HARP+TM in terms of total number of transmitted packet is higher than DSR and OLSR in more stressful situations, most of the packets are due to beaconing process used for topology management. In HARP+TM, a large amount of control packets are broadcasted locally, while in AODV, DSR and OLSR they are

broadcasted globally. When we look at the overhead in bytes, the lowest overhead is incurred by OLSR following by HARP+TM. This is because OLSR does not adapt to increase mobility, while HARP+TM detects and reacts to more link failure when mobility increases. We claim that the proposed architecture and protocols offers promising performance results, and has the best chance to scale to a large network.

Many questions on the proposed architecture remain unanswered. The areas inviting further investigation include:

- What is the best criteria under which the constructed forest maximizes the load balancing in the network ?
- What is the effect of network parameters such as mobility rate, network density and size, number of traffic sources, traffic load; and network models such as traffic pattern and mobility model on the performance of the proposed protocol ?
- How the routing performance is affected by type of routing (source vs. hop-by-hop, zone vs. node level, and their combination) ?
- How the quality of service model can be extended to support metrics at lower layers and what are the metrics ?

Bibliography

[1] *Mobile Ad Hoc Networking (MANET)*, Available at $http://tonnant.itd.nrl.navy.mil/manet/manet\_home.html$. [5, 16]

[2] Charles E. Perkins, *Ad Hoc Networking*, Addison-Wesley, 2001. [6, 12, 97]

[3] J. Jubin and J. D. Tornow, "The DARPA packet radio network protocol," *Proceedings of the IEEE*, 1987. [6, 11]

[4] R. E. Kahn, S. A. Gronemeyer, J. Burchfiel, and R. C. Kunzelman, "Advances in packet radio technology," in *Proceedings of IEEE*, 1978. [6]

[5] J. P. Hubaux, T. Gross, J. Y. Le Boudec, and M. Vetterli, "Toward self-organized mobile ad hoc networks: The Terminodes Project," *IEEE Communication Magazine*, 2001. [6]

[6] B. M. Leiner, D. L. Nielson, and F. A. Tobagi, "Issues in packet radio network design," in *Proceedings of the IEEE*, 1987. [6, 7, 14]

[7] P. Gupta and P. R. Kumar, "The capacity of wireless networks," *IEEE Transactions on Information Theory*, 2000. [7, 80]

[8] M. Grossglauser and D. N. C. Tse, "Mobility increases the capacity of ad-hoc wireless networks," in *Proceedings of INFOCOM*, 2001. [7, 80]

[9] T. Camp, J. Boleng, , and V. Davies, "A survey of mobility models for ad hoc network research," in *Proceedings of Wireless Communication & Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications*, 2002. [7, 72, 80]

[10] D. D. Perkins, H. D. Hughes, and C. B. Owen, "Factors affecting the performance of ad hoc networks," in *Proceedings of the IEEE International Conference on Communications (ICC)*, 2002. [7, 80]

[11] I. Aad, *Quality of service in wireless local area networks*, Ph.D. thesis, Universit´e de Joseph Fourier de Grenoble & INRIA, 2002. [11]

[12] Martha Steenstrup, *Routing in Communication Networks*, Printice Hall, 1995. [12, 13, 14, 97]

[13] E. W. Dijkstra, "A note on two problems in connection with graphs," in *Proceedings of Numerische Math*, 1959. [12, 14]

[14] D. Bertseks and R. Gallager, *Data Networks*, Prentice-Hall, Inc., 1987. [12, 14, 98]

[15] E. M. Rover and C. K. Toh, "A review of current routing protocols for ad hoc mobile wireless networks," *IEEE Personal Communications*, vol. 6, no. 2, pp. 46–55, 1999. [12, 97]

[16] S-J. Lee, M. Gerla, and C-K. Toh, "A simulation study of table-driven and on-demand routing protocols for mobile ad hoc networks," *IEEE Network*, vol. 13, no. 4, pp. 48–54, 1999. [12]

[17] J. Broch, D. A. Maltz, D. B. Johnsin, Y-C. Hu, and J. Jetcheva, "A performance comparison of multi-hop wirless ad hoc network routing protocols," in *Proceedings of MobiCom, Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking*, 1998. [12, 49, 72, 82]

[18] Laura Marie Feeney, "A taxonomy for routing protocols in mobile ad hoc networks," Tech. Rep. T99-07, 1999. [12]

[19] A. U. Shankar, C. Alaettinogly, I. Matta, and K. D-Zieger, "Performance comparison of routing protocols using MaRS: Distance-vector versus link-state," in *Proceedings of ACM Sigmetrics/Performance*, 1992. [13, 14, 97]

[20] J. Moy, "OSPF version 2," 1997, RFC 2178. [14]

[21] J. Strater and B. Wollman, "OSPF modeling and test results and recommendations," Tech. Rep., Mitre 96W0000017, 1996. [14]

[22] L. R. Ford Jr. and D. R. Fulkerson, *Flows in Networks*, Princeton, NJ: Princeton University Press, 1962. [14]

[23] C. Cheng, R. Riley, S. P. R. Kumar, and J. J. Garcia-Luna-Aceves, "A loop-free bellman-ford routing protocol without bouncing effet," in *Proceedings of ACM SIGCOMM '89*, September 1989, pp. 224–237. [14]

[24] C. E. Perkins and P. Bhagwat, "Highly dynamic destination sequenced distance-vector routing (DSDV) for mobile computers," in *Proceedings of SIGCOMM'94*. [14]

[25] S. Murthy and J.J. Garcia-Luna-Aceves, "An efficient routing protocol for wireless networks," *ACM Mobile Networks and Applications Journal*, 1996. [14]

[26] T. Chen and M. Gerla, "Global state routing: A new routing scheme for ad-hoc wireless networks," in *Proceedings of IEEE ICC'98*. [14]

[27] C-C. Chiang, "Routing in clustered multihop, mobile wireless networks with fading channel," in *Proceedings of IEEE SICON*, 1997, pp. 197–211. [14]

[28] A. Iwata, C-C. Chiang, G. Peiand 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*, vol. 17, no. 8, pp. 1369–1379, 1999. [14]

[29] G. Pei, M. Gerla, and X. Hong, "LANMAR: landmark routing for large scale wireless ad hoc networks with group mobility," in *Proceedings of MobiHOC- 1th Workshop on Mobile and Ad Hoc Networking and Computing*, 2000. [14]

[30] C. Adjih, P. jacquet, A. Laouiti, P. Minet, P. Muhlethaler, A. Qayyum, and L. Viennot, "Optimized link state routing protocol," 2003, IETF Draft. [14, 23]

[31] M. Lewis, R. Ogier, and F. Templin, "Topology dissemination based on reverse-path forwarding (tbrpf)," 2003, IETF Draft. [14]

[32] S. Basagni, I. Chlamtac, V. Syrotiuk, and B. Woodward, "A distance routing effect algorithm for mobility (DREAM)," in *MobiCom- The ACM/IEEE International Conference on Mobile Computing and Networking*, 1998. [14]

[33] A. Qayyum, L. Viennot, and A. Laouiti, "Multipoint relaying: An efficient technique for flooding in mobile wireless networks," Tech. Rep., INRIA, 2000, RR-3898. [14, 16, 38, 68]

[34] C. E. Perkins and E. M. Royer, "Ad hoc on-demand distance vector routing," in *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications*, 1999. [14, 23, 52, 61, 65]

[35] David B. Johnson and David A. Maltz, "Dynamic source routing in ad hoc wireless networks," in *Proceedings of Mobile Computing*, 1996. [14, 23, 49, 52, 61, 68, 72]

[36] V. D. Park and M. S. Corson, "A highly adaptive distributed routing algorithm for mobile wireless networks," in *Proceedings of INFOCOM'97*. [14]

[37] E. Gafni and D. Bertsekas, "Distributed algorithms for generating loop-free routes in networks with frequently changing topology," *IEEE Transactions on Communications*, 1981. [14]

[38] C-K. Toh, "A novel distributed routing protocol to support ad-hoc mobile computing," *IEEE International Phoenix Conf. on Computers and Communications*, IPCCC'96. [14]

[39] R. Dube, C. D. Rais, K. Wang, and S. K. Tripathi, "Signal stability based adaptive routing (SSR) for ad-hoc mobile networks," *IEEE Personal Communications*, vol. 4, no. 1, pp. 36–45, 1997. [14]

[40] G. Aggelou and R. Tafazolli, "RDMAR: A bandwidth-efficient routing protocol for mobile ad hoc networks," in *Proceedings of WOWMOM- Workshop on Wireless Mobile Multimedia*, 1999. [14, 16, 52, 68, 70]

[41] Y-B. Ko and N. H. Vaidya, "Location-aided routing in mobile ad hoc networks," *4th Annual International Conference on Mobile Computing and Networking*, MOBICOM'98. [14, 16, 52]

[42] M. R. Pearlman and Z. J. Hass, "Determining the optimal configuration for zone routing protocol," *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 8, pp. 1395–1414, 1999. [15, 24]

[43] M. Joa-Ng and I-Tai Lu, "A peer-to-peer zone-based two-level link state routing for mobile ad hoc networks," *IEEE on Selected Areas in Communications*, vol. 17, no. 8, pp. 1415–1425, 1999. [15, 24, 61]

[44] M. Jiang, J. Li, and Y. C. Tay, "Cluster based routing protocol," IETF Draft, 1999. [15]

[45] B. Williams and T. Camp, "Comparison of broadcasting techniques for mobile ad hoc networks," in *Proceedings of MobiHoc*, 2002. [15, 16, 68]

[46] J. Jetcheva, Y. Hu, D. Maltz, and D. Johnson, "A simple protocol for multicast and broadcast in mobile ad hoc networks," 1001, Internet Draft. [15, 16]

[47] Y.-C. Tseng, S.-Y. Ni, Y.-S. Chen, and J.-P. Sheu, "The broadcast storm problem in a mobile ad hoc network," in *Proceedings of ACM Wireless Networks*, 2002. [15, 16, 68]

[48] W. Peng and X. Lu, "On the reduction of broadcast redundancy in mobile ad hoc networks," in *Proceeding of MobiHoc*, 2000. [16]

[49] David B. Johnson, David A. Maltz, and Yih-Chun Hu, "The dynamic source routing protocol for mobile ad hoc networks (DSR)," 2003, draft-ietf-manet-dsr-08.txt. [16, 68]

[50] C. E. Perkins, E. M. Royer, and S. R. Das, "Ad hoc on-demand distance vector (AODV) routing," draft-ietf-manet-aodv-13, 2003. [16, 65]

[51] R. Castaneda and S. R. Das, "Query localization techniques for on-demand routing protocols in ad hoc networks," in *Proceedings of ACM MobiCom*, 1999. [16]

[52] N. Nikaein and C. Bonnet, "An architecture for improving routing and network performances in mobile ad hoc networks," *To be published in Kluwer/ACM MONET*, 2003. [17]

[53] Navid Nikaein, H. Labiod, and C. Bonnet, "DDR-distributed dynamic routing algorithm for mobile ad hoc networks," in *Proceedings of MobiHOC- First Annual Workshop on Mobile and Ad Hoc Networking and Computing*. IEEE, Augest 2000, pp. 19–27. [17, 23]

[54] N. Nikaein, S. Wu, C. Bonnet, and H. Labiod, "Designing routing protocol for mobile ad hoc networks," in *Proceedings of DNAC2000: 14th conference of New Architectures for Communications*. [17]

[55] N. Nikaein and C. Bonnet, "Improving routing and network performance in mobile ad hoc networks using qulaity of nodes," in *Proceedings of WiOpt'03-xs Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, March 2003. [18, 42]

[56] N. Nikaein, C. Bonnet, and Neda Nikaein, "HARP - hybrid ad hoc routing protocol," in *Proceedings of IST - International Symposium on Telecommunications*, 2001. [18, 48, 60]

[57] N. Nikaein and C. Bonnet, "LQoS- layered quality-of-service model for mobile ad hoc networks," in *Proceedings of SCI*. 6th World Multiconference on Systemics, Cybernetics and Informatics, 2002. [19, 42, 86]

[58] N. Nikaein and C. Bonnet, "A glance at QoS models in mobile ad hoc networks," in *Proceedings of DNAC2002: 16th conference of New Architectures for Communications*, 2002. [19, 86]

[59] S. Wu and C. Bonnet, "DDR-based multicast protocol with dynamic core (DM-PDC)," in *in Procd. of Protocols For High-Speed Networks (PfHSN 2002)*, 2002. [20]

[60] S. Radhakrishnan, N. S. V. Rao G. Racherla, C. N. Sekharan, and S. G. Batsell, "DST - a routing protocol for ad hoc networks using distributed spanning trees," *IEEE Wireless Communications and Networking Conference*, pp. 100–104, 1999. [24]

[61] Z. J. Hass and M. R. Pearlman, "The zone routing protocol (ZRP) for ad hoc networks," Internet Draft, 2000. [24]

[62] P. Jacquet and L. Viennot, "Overhead in mobile ad-hoc network protocols," Tech. Rep., INRIA-RR3965, 2000. [38]

[63] L. Kleinrock and J. Silvester, "Optimum transmission radii for packet radio networks or why six is a magic number," in *Proceedings of the IEEE National Telecommunications Conference*, 1978. [45]

[64] E. Royer, P. Melliar-Smith, and L. Moser, "An analysis of the optimum node density for ad hoc mobile networks," in *Proceedings of the IEEE International Conference on Communication*, 2001. [45]

[65] Feng Xue and P. R. Kumar, "The number of neighbors needed for connectivity of wireless network," 2002, submitted in Wirelesss Networks. [45]

[66] C. Bettstetter, "On the minimum node degree and connectivity of a wireless multihop network," in *Proceedings of MobiHoc*, 2002. [45]

[67] K. Fall and K. Varadhan, *NS notes and documentation*, A Collaboration between researchers at UC Berkeley, LBL, USC/ISI, and Xerox PARC, available at: $http://www.isi.edu/nsnam/ns/$. [49, 93]

[68] D. Cavin, Y. Sasson, and A. Schiper, "On the accuracy of MANET simulators," in *Proceedings of the Workshop on Principles of Mobile Computing (POMC)*, 2002. [49]

[69] M. Takai, J. Martin, and R. Bagrodia, "Effects of wireless physical layer modeling in mobile ad hoc networks," in *Proceedings of MobiHoc*, 2001. [49]

[70] S. R. Das, C. E. Perkins, and E. M. Royer, "Performance comparision of two on-demand routing protocols for ad hoc networks," in *Proceedings of IEEE Infocom*, 2000. [49, 52, 72, 73, 82, 101]

[71] P. Johansson, T. Larsson, N. Hedman, and B. Mielczarek, "Routing protocols for mobile ad-hoc networks - a comparative performance analysis," in *Proceedings of ACM MobiCom*, 1999. [49, 52, 72, 82]

[72] D. S. J. De Couto, D. Aguayo, B. A. Chambers, and R. Morris, "Performance of multihop wireless networks: Shortest path is not enough," . [50]

[73] L. Blazevic, S. Giordano, and J.-Y. Le Boudec, "Self organized terminode routing," *In Journal of Cluster Computing*, 2001. [52]

[74] B. Karp and H. Kung, "GPSR: Greedy perimeter stateless routing for wireless networks," in *Proceedings of MobiCOM- 6th ACM/IEEE International Conference on Mobile Computing and Networking*, 2000. [52]

[75] N. Nikaein, C. Bonnet, N. Akhtar, and R. Tafazolli, "HARPv2 - hybrid ad hoc routing protocol," To be submitted in IEEE Network Magazine. [60]

[76] G. Aggelou and R. Tafazolli, "Determining the optimal configuration for the relative distance microdiscovery ad hoc routing protocol," *IEEE Transaction on Vehicular Technology*, 2002. [68]

[77] B. Tuch, "Development of wavelan, an ism band wireless lan," *AT&T Technical Journal*, vol. 72, no. 4, pp. 27–33, July/August 1993. [72]

[78] J. Yoon, M. Liu, and B. D. Noble, "Random waypoint considered harmful," in *Proceedings of InfoCom*, 2003. [72]

[79] R. Knopp and P. A. Humblet, "Information capacity and power control in single-cell multiuser communications," in *Proceedings of Int. Conf. Communications*, 1995. [80]

[80] T. Clausen, P. Jacquet, and L. Viennot, "Comparative study of routing protocols for mobile ad hoc networks," in *Med-hoc-Net*, 2002. [83]

[81] E. Crawley, R. Nair, B. Rajagopalan, and H. Sandick, "A framework for QoS-based routing in the internet," Aug. 1998, RFC 2386. [86]

[82] A. J. Goldsmith and S. B. Wicker, "Design challanges for energy-constrained ad hoc wireless networks," *IEEE Wireless Communications*, 2002. [86]

[83] P. Mohapatra, J. Li, and C. Gui, "QoS in mobile ad hoc networks," *IEEE Wireless Communications Magazine*, 2002, to appear. [87]

[84] S. Chen and K. Nahrstedt, "Distributed quality-of-service routing in ad hoc networks," *IEEE Journal on Selected Areas in Communication*, vol. 17, no. 8, 1999. [87]

[85] D. D. Perkins and H. D. Hughes, "A survey on quality-of-service support for mobile ad hoc networks," *Wireless Communications and Mobile Computing*, 2002. [87]

[86] D. Chalmers and M. Sloman, "A survey of QoS in mobile computing environments," *IEEE Communications Surveys*, 1999. [87]

[87] X. Xiao and L. M. Ni, "Internet QoS: A big picture," *IEEE Network*, pp. 8–18, Match/April 1999. [87]

[88] R. Braden, D. Clark, and S. Shenker, "Integrated services in the Internet architecture: an overview," 1994, IETF RFC 1633. [87]

[89] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An architecture for differentiated services," 1998, IETF RFC 2475. [87]

[90] H. Xiao, W. K. G. Seahand A. Lo, and K. C. Chua, "A flexible quality of service model for mobile ad-hoc networks," in *Proceedings of IEEE VTC2000-spring*, Japon/Tokyo, 2000. [87, 88]

[91] Lixia Zhang, Stephen Deering, and Deborah Estrin, "RSVP: A new resource reservation protocol," *IEEE network*, 1993. [87]

[92] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin, "Resource reservation protocol (RSVP) – version 1 functional specification," Sept. 1997, IETF RFC 2205. [87]

[93] S. Shenker, C. Partridge, and R. Guerin, "Specification of guaranteed quality of service," 1997, RFC 2212. [87]

[94] J. Wroclawski, "Specification of the controlled-load network element service," 1997, RFC 2211. [87]

[95] Neda Nikaein, *QoS Control in Wireless IP Access Networks*, Ph.D. thesis, ENST, Sophia-Antipolis - France, October 2000. [91]

[96] A. Demers, S. Keshav, and S. Shenker, "Analysis and simulation of a fair queueing algorithm," in *Proceedings of the Sigcom'89, Symposium on Communications Architecture and Protocols*, 1989. [96]

[97] D. Clark, S. Shenker, and L.Zhang, "Supporting real-time applications in an integrated service packet networks: Architecture and mechanism," in *Proceedings of ACM SIGCOM'92*, 1992. [96]

[98] R.E. Bellman, *Dynamic Programming*, Princeton Univ. Press, 1957. [98]

Glossary

ALM	Application Layer Metrics
CBR	Constant Bit Rate
CB-WFQ	Class-Based Weighted Fair Queuing
DARPA	Defense Advanced Research Projects Agency
DBF	Distributed Bellman-Ford
DSCP	DiffServ Code Point
GAN	Group Area Network
GPS	Global Positioning System
IETF	Internet Engineering Task Force
LUT	Last Update Time
MANET	Mobile Ad Hoc Network
NID	Node Identifier
NLM	Network Layer Metrics
PAN	Personal Area Network
PERR	Path Error Message
PHB	Per-Hop Behavior
PN	Preferred Neighbor
PREQ	Path Request Message
PREP	Path Reply Message
QoC	Quality of Connectivity
QoS	Quality of Service
RDE	Relative Distance Estimation
RDM	Relative Distance Microdiscovery
RSVP	Resource ReSerVation Protocol
SP	Shortest Path
TTL	Time-to-Live

List of Tables

II-1 Comparison of HARP with the IETF routing protocols 20

III-1 Neighboring table 25
III-2 Intra-zone table 26
III-3 Inter-zone table 26
III-4 Neighboring table of node k 29
III-5 Intra-zone table of nodes k and f regarding Fig. III-4 30
III-6 Intra-zone table of nodes k and f regarding Fig. III-4 32
III-7 Inter-zone table of node d 33

IV-1 Routing table entry 62
IV-2 Intra-zone table of nodes k, f and c regarding Fig. IV-2(a) 63

V-1 QoS Classes & Mapping 92

List of Figures

III-1 Fields of a beacon 25

III-2 Diagram illustrating the different phases of the algorithm DDR 27

III-3 Each node in the graph is characterized by its degree and a letter which represents its ID number, and we assume that each node knows the ID numbers and the degrees of its neighboring nodes 28

III-4 Constructed forest 29

III-5 Constructed forest with minimum neighborhood degree 31

III-6 View of node k on its tree 32

III-7 Zone naming 34

III-8 Zone partitioning (reduced graph) 35

III-9 The proof of Theorem 1 37

III-10 Flow chart of beacons in forward modes: a beacon arrives at a node x from a node y in PN Forward mode 40

III-11 Flow chart of beacons in election modes: a beacon arrives at a node x from a node y in PN Elect mode 41

III-12 Number of edges used for routing as a function of number of nodes in a flat network and in the forest 46

III-13 Comparison of constant and variable density of zone partitioning methods 47

III-14 Packet delivery fraction for the 50 nodes with various number of sources 51

III-15 Average data packet delay for the 50 nodes with various number of sources 53

III-16 Routing overhead in terms of number of packets and bytes for 50 nodes with various number of sources 54

III-17 Routing overhead in terms of number of packets for the 50 nodes with various number of sources 55

III-18 DDR Architecture 56

IV-1 Intra-zone routing 63

IV-2 Inter-zone routing 65

IV-3 Relative distance estimation 70

IV-4 Relative distance estimation 71

IV-5 Packet delivery fraction for the 50 nodes with various number of sources 75

IV-6 Average data packet delay for the 50 nodes with various number of sources 76

IV-7 Routing overhead in terms of total number of transmitted packets for the 50 nodes with various number of sources 78

IV-8 Routing overhead in terms of total transmitted bytes for the 50 nodes with various number of sources 79

IV-9 The effect of traffic load on routing performance for high and no mobility rate 81

V-1 Global view of layered quality of service model 90

V-2 Packet delivery fraction for the 50 nodes with various number of sources 94

V-3 Service Differentiation in an Ad hoc Node 96

Curriculum Vitae

Navid NIKAEIN was born in Tehran, Iran. He received the B.Sc. degree in Mathematics Applied to Computer Science in 1998 from the Faculty of Mathematics of Shahid-Beheshti University in Tehran. Then, he obtained the M.Sc. degree in Networking and Distributed Systems from University of Nice Sophia-Antipolis (ESSI), France; and his Master Thesis on Mobile Agents Systems at INRIA - Meije project, Sophia-Antipolis. He joined Institut Eur´ecom in December 1999 as a Ph.D. student under the supervision of Christian Bonnet, enrolled at Ecole Polytechnique Federale de Lausanne (EPFL), Switzerland. In 2003, he has been visiting CCSR, Centre for Communication Systems Research, Surrey University, England, as a researcher for a period of four months. His current research interests include routing, quality of service, and mobility management issues in mobile environements.

List of Publications

- N. Nikaein, H. Labiod, C. Bonnet, “DDR - Distributed Dynamic Routing Algorithm for Mobile Ad Hoc Network”, *Proceedings of MobiHoc*, Boston, USA, 2000.
- N. Nikaein, S. Wu, C. Bonnet, H. Labiod,“Designing Routing Protocol for Mobile Ad Hoc Networks”, *Proceedings of DNAC*, Paris, France, 2000.
- N. Nikaein, C. Bonnet, Neda Nikaein, “HARP - Hybrid Ad Hoc Routing Protocol”, *Proceedings of IST*, Tehran, Iran, 2001.
- N. Nikaein, C. Bonnet, Y. Moret, I. A. Rai, ”2LQoS - Two-Layered Quality of Service Model for Reactive Routing Protocols for Mobile Ad Hoc Networks”, *Proceedings of SCI*, Orlando, USA, 2002.
- N. Nikaein, C. Bonnet,“A Glance at QoS Models in Mobile Ad Hoc Networks”, *Proceedings of DNAC*, Paris, France, 2002.
- N. Nikaein, C. Bonnet,”ALM - Adaptive Location Management Model Incorporating Fuzzy Logic For Mobile Ad Hoc Networks”, *Proceeding of med-hoc-net*, Sardegna, Italy, 2002.
- N. Nikaein, C. Bonnet,“Improving Routing and Network Performance in Mobile Ad Hoc Networks using Quality of Nodes”, *Proceedings of WiOpt’03*, Sophia-Antipolis, France, 2003.
- N. Nikaein, C. Bonnet,“Topology Management for Improving Routing and Network Performances in Mobile Ad Hoc Networks”, *Kluwer/ACM MONET Journal*, to be published, 2003.
- N. Nikaein, C. Bonnet, N. Akhtar, R. Tafazolli, “HARPv2 - Hybrid Ad Hoc Routing Protocol”, To be submitted.