SCALABLE ROUTING PROTOCOLS WITH APPLICATIONS TO MOBILITY

THÈSE N^{O} 2517 (2002)

PRÉSENTÉE À LA FACULTÉ IC SECTION SYSTÈMES DE COMMUNICATION

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

POUR L'OBTENTION DU GRADE DE DOCTEUR ÈS SCIENCES TECHNIQUES

PAR

Ljubica BLAZEVIC

B.Sc. in Electrical Engineering, University of Belgrade, Yougoslavie
et de nationalité yougoslave

acceptée sur proposition du jury:

Prof. J.-Y. Le Boudec, directeur de thèse
Prof. C. Bonnet, rapporteur
Prof. J. Crowcroft, rapporteur
Dr P. Rouzet, rapporteur
Prof. N. Vaidya, rapporteur

Lausanne, EPFL
2002

SCALABLE ROUTING PROTOCOLS WITH APPLICATIONS TO MOBILITY

by

Ljubica Blažević

Abstract

In this thesis we study two routing problems related to mobility. The first problem concerns scalable multicast routing when there is a large number of multicast groups with a small number of receivers. Existing dense and sparse mode routing protocols have the following problems when the number of groups is large, but the groups are small: (1) high overhead in control traffic (dense mode), (2) poor utilization of backbone links and (3) complexity of management of single core routers (sparse mode). Our solution to this problem is the Distributed Core Multicast (DCM) routing protocol. DCM is an extension of the core-based tree approach and its architecture is based on several core routers per multicast groups. We explain the objectives achieved with DCM: (1) avoiding state information in backbone routers, (2) avoiding triangular routing across expensive backbone links, (3) scaling well with the number of multicast groups. DCM can be applied to support host micromobility in a large single Internet domain network, where every mobile host is assigned a multicast address in a domain it visits. The advantages of the multicasting-based approach for supporting host mobility are low latency and minimal disruption during handover.

The second problem studied in this thesis concerns scalable routing in a large scale mobile ad hoc network. Ad hoc networks have gained increasing popularity in recent years because of their ease of deployment. No wired base station or infrastructure is needed, and mobile nodes communicate with each other using multi-hop wireless links. In ad hoc networks, routing protocols are challenged by the establishment of and the maintenance of multihop routes in the face of mobility, bandwidth limitation and power constraints. Routing protocols that rely on state concerning all links on the network, or all links on a route between a source and a destination result in poor scaling properties in larger mobile ad hoc networks. Position-based routing protocols are better suited for large mobile ad hoc networks. These protocols use the positions of nearby nodes and of packet's destination to make the packet forwarding decisions. We present a scalable routing protocol for a large mobile ad hoc network that is called terminode network. We call the nodes terminodes because they act as network nodes and terminals at the same time. Our routing scheme is a combination of two protocols called Terminode Local Routing (TLR) and Terminode Remote Routing (TRR). TRR is activated when the destination is remote and it uses the location of the destination obtained either via location management or by location tracking. TLR acts when the packet gets close to the destination and it uses routing tables that are built by nodes for close terminodes. The use of TRR results in a scalable solution that reduces dependence on the intermediate systems, while TLR allows to increase the probability of reaching the destination, even when it has moved considerably from the location that was known at the source. We

use simulations to demonstrate terminode routing’s scalability in different sized mobile ad hoc networks.

Version Abrégée

Dans ce travail de thèse, nous étudions deux problèmes de routage liés la mobilité. Le premier concerne le routage multicast à grande échelle, pour un grand nombre de groupes multicast avec un petit nombre d'utilisateurs. Dans ce cas, les protocoles existants en mode dense ou éparpillé ont les problèmes suivants: (1) une surcharge importante de trafic de contrôle (pour les protocoles en mode dense), (2) une faible utilisation du réseau d'épine dorsale et (3) une complexe gestion des routeurs individuels du noyau (pour les protocoles en mode éparpillé). Notre solution ce problème est l'algorithme de routage appelé Multicast Distribué dans le Noyau (DCM: Distributed Core Multicast). DCM est une extension des algorithmes basés sur un arbre dans le noyau, et son architecture utilise plusieurs routeurs du noyau par groupe multicast. Nous expliquons les objectifs atteints avec DCM, savoir (1) éviter de stocker de l'information d'état dans les routeurs du réseau d'épine dorsale, (2) éviter des routages triangulaires entre les liens coûteux du réseau d'épine dorsale et (3) supporter un grand nombre de groupes multicast. DCM peut supporter une micromobilité des hôtes l'intérieur d'un grand Internet domaine unique, oú chaque hôte mobile reçoit une adresse multicast dans le domaine qu'il visite. Les avantages d'une approche basée sur le multicast pour supporter la mobilité des hôtes sont un faible temps de latence et une interruption du service minimale pendant les transfers de cellule.

Le second problème étudié dans cette thèse est celui du routage performant grande échelle dans des grands réseaux mobiles ad-hoc. Depuis quelques années, les réseaux ad-hoc connaissent une forte popularité cause de leur facilité de déploiement. Aucune station de base, ni infrastructure fixe, n'est nécessaire; les noeuds mobiles communiquent entre eux en passant par des relais intermédiaires. La communication entre les deux noeud emprunte donc plusieurs liens sans fils. L'établissement et la maintenance de telles routes posent de nouveaux défis aux protocoles de routage dans les réseaux ad-hoc, confrontés la mobilité, ainsi qu'aux limitations de capacité des liens et de puissance des noeuds. Les protocoles basés sur la position des noeuds sont mieux adaptés aux réseaux de grande taille que ceux se basant sur l'état de chaque lien du réseau complet, ou même d'une route particulière. Nous présentons un protocole de routage performant pour des réseaux mobiles ad hoc de grande taille, appelés réseaux de terminodes. On appelle les noeuds, terminodes, parce qu'ils accumulent la fonctionnalité des noeuds du réseau avec celle des terminaux. Notre protocol est une combinaison des deux protocoles appelés Routage Local des Terminodes (TLR: Terminode Local Routing) et Routage Distance des Terminodes (TRR: Terminode Remote Routing). Ce dernier est activé quand la destination est éloignée, et utilise sa position géographique obtenue soit par un algorithme de gestion de localisation, ou de poursuite. Le premier est activé quand un paquet arrive proximité de la destination et utilise des tables

de routage construites par des terminodes proches. L'utilisation de TRR réduit la dépendance du routage des systèmes intermédaires, tandis que TLR augmente la probabilité d'atteindre la destination même si cette dernière s'est déplacée considérablement de sa position connue par la source. Nous démontrons par simulation les capacités d'adaptation de ce protocole de routage des réseaux de grande taille.

To my parents.

Acknowledgements

First and foremost I want to thank Prof. Jean-Yves Le Boudec who was an excellent thesis advisor. I am deeply grateful to him for the time he spent guiding and contributing to this work.

I would like to thank all my colleagues at the Laboratory for computer Communications and their Applications (LCA) for making our lab a very pleasant and friendly place to work at: Catherine Boutremans, Olivier Dousse, Paul Hurley, Božidar Radunović, Patrick Thiran, Milan Vojnović, Levente Buttyan, Mario Čagalj, Srdjan Čapkun, Jun Luo, Naouel Ben Salem. I am grateful to the LCA system managers: Marc-André Luthi and Jean-Pierre Dupertuis who provided me with the stable computing environment. I also want to thank our secretaries Danielle Alvarez, Holly Cogliati and Angela Devenoge for making things a lot easier.

I spent a particularly enjoyable two years while working in the Terminodes project. I am grateful to Silvia Giordano for many fruitful discussions during this project.

I would like to thank Chadi Barakat for his help in the last period of my thesis. A special thank you goes out to Sonja Buchegger from IBM in Zurich and Holly Cogliati for reading drafts of my thesis.

There are many students who worked with me and helped a lot in developing things. Among them I especially thankful to Ivan Mitev and Alok Agarwal.

I am grateful to my friends from Lausanne: Cane Čekerevac, Branko Glišić, Jelena Godjevac, Sandra and Dragan Manić, Bojana and Zoran Randjelović, Božidar Radunović, Bojana and Ljubiša Misković, Jelena Debljević. We had a great time together.

Finally, my gratitude goes to my parents Ljiljana and Dragan for their love, support and encouragement while working on the thesis.

Contents

List of Figures

Chapter 1

Introduction

In this thesis we are interested in routing. In the first phase we have considered scalable multicast routing when there is a large number of multicast groups with a small number of receivers. This problem is studied in the fixed Internet.

In the second phase we have considered scalable routing in a large mobile ad hoc network.

Multicast routing for many small multicast groups

The first phase of the thesis concerns multicast routing for many small multicast groups. This occurs when the number of multicast groups is very large (e.g., greater than a million), the number of receivers per multicast group is very small (e.g., less than five) and each host is a potential sender to a multicast group. For such cases, existing dense or sparse mode multicast routing algorithms do not scale well with the number of multicast groups.

Dense mode multicast routing protocols, such as DVMRP[84] and MOSPF[51], are intended for use within regions where multicast groups are densely populated or bandwidth is plentiful. Both protocols use source specific shortest path trees. These routing schemes require that each multicast router in the network keeps per source per group information and therefore routers have to keep a large amount of multicast group state information. In addition, dense mode routing protocols are subject to high routing overhead in the case of sparse multicast groups.

Existing sparse mode multicast routing protocols, such as the protocol independent multicast (PIM-SM) [17] and the core-based trees (CBT) [5], build a single delivery tree per multicast group that is shared by all senders in the group. This tree is rooted at a single centre router called "core" in CBT, and "rendezvous point" (RP) in PIM-SM.

Both centre-based routing protocols have the following potential shortcomings:

- traffic for the multicast group is concentrated on the links along the shared tree, mainly near the core router;

- finding an optimal centre for a group is a NP-complete problem and requires the knowledge of the whole network topology. Data distribution through a single centre router could cause non-optimal distribution of traffic in the case of a bad positioning of the centre router, with respect to senders and receivers. This problem is known as a triangular routing problem.

In this thesis we propose the Distributed Core Multicast (DCM) routing protocol as a solution for scalable multicast with many small multicast groups.

DCM is based on an extension of the centre-based tree approach and is designed for the efficient and scalable delivery of multicast data under the assumptions that we mention above (a large number of multicast groups, a few receivers per group and a potentially large number of senders to a multicast group).

The following is a short overview of DCM. We consider a network model where a large single domain network is configured into areas that are organized in a two-level hierarchy. At the top level is a single backbone area. All other areas are connected via the backbone. The issues addressed by DCM are: (1): to avoid multicast group state information in backbone routers, (2): to avoid triangular routing across expensive backbone links and (3) to scale well with the number of multicast groups.

In our solution, we introduce an architecture based on several core routers per multicast group, called Distributed Core Routers (DCRs).

- The DCRs in each area are located at the edge of the backbone. The DCRs act as backbone access points for the data sent by senders inside their area to receivers outside this area. A DCR also forwards the multicast data received from the backbone to receivers in the area it belongs to. When a host wants to join the multicast group *M*, it sends a join message. This join message is propagated hop-by-hop to the DCR inside its area that serves the multicast group. Conversely, when a sender has data to send to the multicast group, it will send the data encapsulated to the DCR assigned to the multicast group.

- The Membership Distribution Protocol (MDP) runs between the DCRs serving the same range of multicast addresses. It is fully distributed. MDP enables the DCRs to learn about other DCRs that have group members.

- The distribution of data uses a special mechanism between the DCRs in the backbone area, and the trees rooted at the DCRs towards members of the group in the other areas.

We propose a special mechanism for data distribution between the DCRs, which does not require that non-DCR backbone routers perform multicast routing.

The benefits of the introduction of the DCRs close to any sender and receivers are the following:

- converging traffic is not sent to a single centre router in the network,
- data sent from a sender to a group within the same area is not forwarded to the backbone,
- the triangular routing problem common to all centre-based trees is alleviated and, unlike PIM-SM, is suitable for groups with many sporadic senders,
- similar to PIM-SM and CBT, DCM is independent of underlying unicast routing protocol.

We have examined the properties of DCM in a large single domain network. DCM is implemented using the Network Simulator (NS) tool. Our simulation results tend to indicate that DCM performs better than existing, sparse mode, routing protocols in terms of multicast forwarding table size.

We give an example of the application of DCM where it is used to support host micromobility in a large single domain network. In this solution every mobile host is assigned a multicast address in a domain it visits. DCM-based mobility management is not an alternative to Mobile IP [57] because DCM is not a solution to wide-area mobility. In contrast, this approach can be used as an alternative to Cellular IP [82] within a single domain network. We advantages of the multicasting-based approach for supporting host mobility are low latency and minimal disruption during handover.

Large scale mobile ad hoc routing: Terminode routing

In the second phase of this thesis we focus on the problem of unicast routing in a large mobile ad hoc network, called terminode network. We call the nodes $terminodes$ because they act as network nodes and terminals at the same time.

An ad hoc network is a collection of wireless mobile nodes dynamically forming a temporary network without the use of any existing network infrastructure or centralized administration. Each node can communicate directly with other nodes within its transmission range, they are called neighbours. In order to send data to a non-neighbour destination, a node sends data first to its neighbour which in turn sends data to its neighbour, and so on until the destination is reached.

The routing problem is the problem of finding a route for sending data from a source to a given destination. Recently a variety of routing protocols has been proposed for such mobile, wireless networks. The existing routing protocols can be classified either as proactive or reactive.

Proactive protocols attempt to maintain routes continuously, so that the route is already available when it is needed for a packet to be forwarded. In those protocols, routing tables are exchanged among neighbouring nodes each time a change occurs in the network topology. As a consequence, proactive protocols are not suitable when the mobility rate in the network is high.

An attempt to overcome these limitations is to look for a route only on demand. This is the basic idea of reactive protocols. In reactive protocols a control message is sent to discover a route to a given destination. Reactive protocols have lower control traffic overhead than proactive protocols. However, since a route has to be discovered before the actual transmission of the data, these protocols can have a longer delay. Furthermore, due to mobility, the discovered route may be unusable because some links of the route may be broken.

There is a family of routing protocols that base routing on the geographical positions of nodes. Here the source needs to know the geographical position of the destination to which it wishes to send, and it labels packets with the corresponding destination position. An intermediate node only needs to know its own position and the positions of nearby nodes; this is enough information to relay each packet through the neighbour that is geographically closest to the ultimate destination. Because nodes only need local information, regardless of the total network size, geographic forwarding is attractive for large-scale networks.

Terminode routing We designed routing in a terminode network based on the following objectives:

- Routing should scale well in terms of the number of nodes and geographical coverage;
- Routing should have scalable mechanisms that cope with the dynamicity in the network due to mobility;
- Nodes need to be highly cooperative and redundant, but most of all, cannot use complex algorithms or protocols.

Our routing scheme is a combination of two protocols called Terminode Local Routing (TLR) and Terminode Remote Routing (TRR). TRR is used to send data to remote destinations and uses the location information; it is the key element for achieving scalability and reduced dependence on intermediate systems. TLR is a mechanism that allows for destinations to be reached in the

vicinity of a terminode. TLR does not use locations for packet forwarding, and it is a strategy for handling the destination location deviation due to mobility.

We assume that the source terminode knows or can obtain the location of the destination that is used for TRR by means of a location management method. Location management is out of the scope of this thesis.

TRR's default packet forwarding method is *Geodesic Packet Forwarding (GPF)*. GPF is basically a greedy method that forwards the packet closer to the destination location until the destination is reached. GPF does not perform well if the source and the destination are not well connected along the shortest geodesic path.

In order to circumvent large holes in terminode distribution, TRR primarily forwards packets on *anchored paths*. In contrast with traditional routing algorithms, an anchored path does not consist of a list of nodes to be visited to reach the destination. An anchored path is a list of fixed geographic points, called $anchors$. In traditional paths made of lists of nodes, if nodes move far from where they were at the time when the path was computed, the path cannot be used to reach the destination. Given that geographic points do not move, the advantage of anchored paths is that an anchored path is always "valid". In order to forward packets along an anchored path, TRR uses the novel method called *Anchored Geodesic Packet Forwarding (AGPF)*. AGPF is a source loose routing method designed to be robust for mobile networks. With AGPF the packet is sent in the direction of an anchor, thus trying to reach some terminode in proximity of this anchor. Thereon, the packet is forwarded in the direction of the next anchor on the anchored path, and so on. When the packet comes close to the destination, the packet forwarding method becomes TLR.

Anchored paths are obtained at the source by the path discovery methods. We propose two such methods: the first is called *Friend Assisted Path Discovery (FAPD)*, and the second is called Geographic Maps-based Path Discovery (GMPD). FAPD enables the source to learn the anchored path(s) to the destination using, so-called, $friends$, terminodes where the source already knows how to route packets. GMPD is another method for anchored path discovery, which assumes that the network topology is known to all nodes in the network.

In this thesis we focus mainly on single path terminode routing. We also describe the architecture of multipath terminode routing. However, implementation and analysis of multipath terminode routing is outside the scope of this thesis.

The combination of TLR and TRR has the following features:

1. It is highly scalable because every node relies only on itself and a small number of other nodes for packet forwarding;

2. It acts and reacts well to the dynamics of the network because TRR paths are more robust to mobility than the traditional paths made of lists of nodes. By using TLR, routing is more robust to the destination location deviation due to mobility.

3. It can be implemented and run in very simple devices because the algorithms and protocols are simple and based on high cooperation.

Thesis Contents

This thesis is organized as follows:

In Chapter 2, the problem of multicast routing for many small multicast groups is considered. First, the review of existing multicast routing protocols is given, as well as an explanation of why they do not scale well when there are many small multicast groups. Then we describe our solution to this problem, called the Distributed Core Multicast Protocol (DCM). Architecture and protocol specifications of DCM are given. DCM is implemented in NS. Evaluation of DCM is given with the following metrics: amount of multicast router state information, traffic concentration and control traffic overhead. Then, we give examples of the application of DCM: in distributed simulations; in supporting Internet host mobility, and in cellular Internet Telephony.

Beginning with Chapter 3, we consider routing issues in mobile ad hoc networks. In Chapter 3 different routing strategies, found in the literature, are reviewed. This includes the traditional MANET-like routing protocols (that do not use geographic positions) and a class of geographic position-based routing protocols.

In Chapter 4 we describe Terminode Routing, our solution for scalable large area mobile ad hoc routing. Complete descriptions of its components (Terminode Local Routing (TLR) and Terminode Remote Routing(TRR)) are given.

In Chapter 5 we describe the implementation of terminode routing in GloMoSim. A performance evaluation of TLR and TRR is given.

In Chapter 6 we describe avenues for further extension of the work in this thesis.

Chapter 7 concludes this thesis.

Claims

We can summarize the contributions of this thesis as follows:

- We designed a new routing protocol called Distributed Core Multicast (DCM). This protocol scales better than existing multicast routing protocols in the case of many small multicast groups.
- DCM is implemented and evaluated using the Network Simulator (NS).
- We showed how DCM can be applied as a routing protocol to support Internet host mobility within a single domain network.
- We studied the scalable routing problem in a large mobile ad hoc network, and we designed the Terminode Routing protocol. It is the combination of two protocols: Terminode Local Routing (TLR) and Terminode Remote Routing (TRR). TRR uses geographical positions for packet forwarding and it is a key element for achieving scalability. TLR is a mechanism to deal with problems due to position inacurracy. We described the two protocols and the interaction between them.
- Terminode routing is implemented and evaluated in GloMoSim. For the purpose of evaluation we designed the new mobility model, called restricted random waypoint, that better represents the mobility pattern in a realistic large scale mobile adhoc environment.
- Evaluation of the terminode routing performance showed that it scales well in a large mobile ad hoc network composed of several hundred mobile hosts.

Chapter 2

Distributed Core Multicast (DCM)

2.1 Introduction

In this chapter we describe a multicast routing protocol called Distributed Core Multicast (DCM). DCM is designed to provide low overhead delivery of multicast data in a large single domain network for a very large number of small groups. This occurs when the number of multicast groups is very large (e.g., greater than a million), the number of receivers per multicast group is very small (e.g., less than five) and each host is a potential sender to a multicast group.

DCM is a sparse mode routing protocol, designed to scale better than the existing multicast routing protocols when there are many multicast groups, but each group has in total a few members.

Relevant aspects of existing multicast routing protocols are described in Section 2.2. Sparse mode multicast routing protocols, such as the protocol independent multicast (PIM-SM) [17] and the core-based trees (CBT) [5], build a single delivery tree per multicast group that is shared by all senders in the group. This tree is rooted at a single centre router called "core" in CBT, and "rendezvous point" (RP) in PIM-SM.

Both centre-based routing protocols have the following potential shortcomings:

- traffic for the multicast group is concentrated on the links along the shared tree, mainly near the core router;

- finding an optimal centre for a group is a NP-complete problem and requires the knowledge of the whole network topology [87]. Current approaches typically use either an administrative selection of centers or a simple heuristic [76]. Data distribution through a single centre router could cause non-optimal distribution of traffic in the case of a bad position-

ing of the centre router, with respect to senders and receivers. This problem is known as a triangular routing problem.

PIM-SM is not only a centre-based routing protocol, but it also uses source-based trees. With PIM-SM, destinations can start building source-specific trees for sources with a high data rate. This partly addresses the shortcomings mentioned above, however, at the expense of having routers on the source-specific tree keep source-specific state. Keeping the state for each sender is undesirable when the number of senders is large.

DCM is based on an extension of the centre-based tree approach and is designed for the efficient and scalable delivery of multicast data under the assumptions that we mention above (a large number of multicast groups, a few receivers per group and a potentially a large number of senders to a multicast group).

As a first simplifying step, we consider a network model where a large single domain network is configured into areas that are organized in a two-level hierarchy. At the top level is a single backbone area. All other areas are connected via the backbone(see Figure 2.1). This is similar to what exists with OSPF[52].

The issues addressed by DCM are: (1): to avoid multicast group state information in backbone routers, (2): to avoid triangular routing across expensive backbone links and (3) to scale well with the number of multicast groups.

The following is a short DCM overview and it is illustrated in Figure 2.1. We introduce an architecture based on several core routers per multicast group, called Distributed Core Routers (DCRs).

- The DCRs in each area are located at the edge of the backbone. The DCRs act as backbone access points for the data sent by senders inside their area to receivers outside this area. A DCR also forwards the multicast data received from the backbone to receivers in the area it belongs to. When a host wants to join the multicast group *M*, it sends a join message. This join message is propagated hop-by-hop to the DCR inside its area that serves the multicast group. Conversely, when a sender has data to send to the multicast group, it will send the data encapsulated to the DCR assigned to the multicast group.

- The Membership Distribution Protocol (MDP) runs between the DCRs serving the same range of multicast addresses. It is fully distributed. MDP enables the DCRs to learn about other DCRs that have group members.

- The distribution of data uses a special mechanism between the DCRs in the backbone area, and the trees rooted at the DCRs towards members of the group in the other areas.

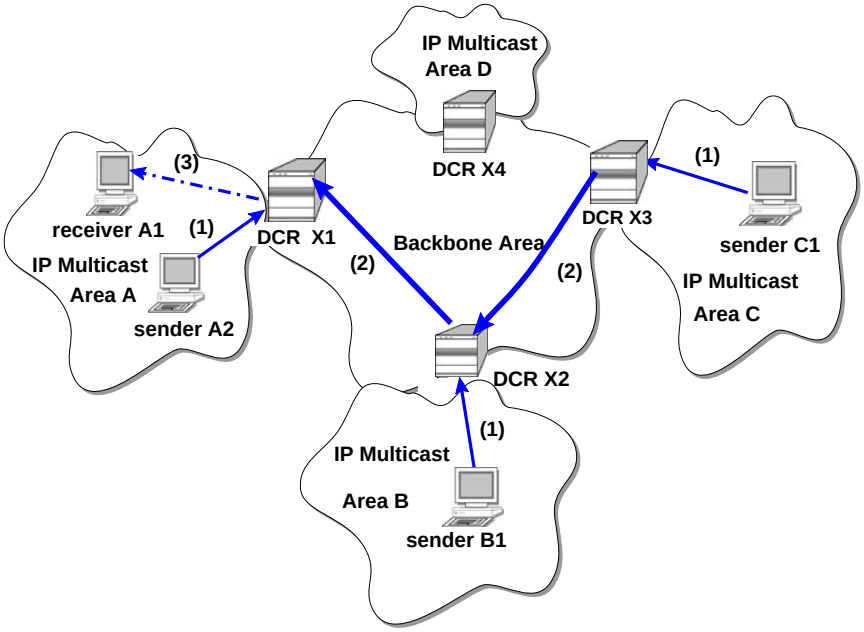


Figure 2.1: This is a model of a large single domain network and an overview of data distribution with DCM. In this example there are four non-backbone areas that communicate via the backbone. We show one multicast group *M* and DCRs X1, X2, X3 and X4 that serve *M*. Step (1): Senders A2, B1 and C1 send data to the corresponding DCRs inside their areas. Step (2): DCRs distribute the multicast data across the backbone area to DCR X1 that needs it. Step (3): A local DCR sends data to the local receivers in its area.

> We propose a special mechanism for data distribution between the DCRs, which does not require that non-DCR backbone routers perform multicast routing.

With the introduction of the DCRs close to any senders and receivers, converging traffic is not sent to a single centre router in the network. Data sent from a sender to a group within the same area is not forwarded to the backbone. Our approach alleviates the triangular routing problem common to all centre-based trees, and unlike PIM-SM, is suitable for groups with many sporadic senders. Similar to PIM-SM and CBT, DCM is independent of underlying unicast routing protocol.

In this chapter we examine the properties of DCM in a large, single domain network. However, DCM is not constrained to a single domain network. Interoperability of DCM with other inter-domain routing protocols is the object of future work.

The structure of this chapter is as follows. In the next section we give an overview of the existing multicast routing protocols. In Section 2.3 we present the architecture of DCM. That is followed by the DCM protocol specification in Section 2.4. In Section 2.5 we give the evaluation

of DCM. Sections 2.7 and 2.8 present different applications of DCM.

2.2 Overview of Multicast Routing Protocols

There are two basic families of algorithms that construct multicast trees used for the distribution of IP multicast data: source specific trees and group shared trees. In the former case an implicit spanning tree per source is calculated, which is minimal in terms of transit delay from a source to each of the receivers . In the latter case only one shared tree, which is shared by all sources, is built. There are two types of shared trees. One type is the Steiner minimal tree (SMT)[89]. The main objective is to build a tree that spans the group of members with a minimal cost and thus globally optimise the network resources. Since the Steiner minimal tree problem is NP-complete, numerous heuristics have been proposed [86]. No existing SMT algorithms can be easily applied in practical multicast protocols designed for large scale networks [87]. The other type of shared trees is a centre-based tree that builds the shortest path tree rooted "in the centre" of the networks and spans only receivers of the multicast group.

Below we briefly describe existing dense and sparse mode multicast routing protocols in the Internet.

2.2.1 Dense Mode Multicast Routing Protocols

Traditional multicast routing mechanisms, such as DVMRP[84] and MOSPF[51], are intended for use within regions where multicast groups are densely populated or bandwidth is plentiful. Both protocols use source specific shortest path trees. These routing schemes require that each multicast router in the network keeps per source per group information.

DVMRP is based on the Reverse Path Forwarding (RPF) algorithm that builds a shortest path sender-based multicast delivery tree. Several first multicast packets transmitted from a source are broadcasted across the network over links that may not lead to the receivers of the multicast group. Then the tree branches that do not lead to group members are pruned by sending prune messages. After a period of time, the prune state for each (source, group) pair expires and reclaims the stale prune state. Subsequent datagrams are flooded again until branches that do not lead to group members are pruned again. This scheme is currently used for Internet multicasting over the MBONE.

In MOSPF, together with the unicast routing information, group membership information is flooded so that all routers can determine whether they are on the distribution tree for a particular source and group pair. MOSPF is designed atop a link-state unicast routing protocol called

OSPF[52]. With MOSPF, in order to scale better, a large routing domain can be configured into areas connected via the backbone area. Multicast routers in non-backbone areas have the complete membership information inside their corresponding areas, while the aggregate membership information of the area is inserted in the backbone. Like DVMRP, MOSPF has a high routing message overhead when groups are sparsely distributed.

2.2.2 Sparse Mode Multicast Routing Protocols

Core Based Trees (CBT) Sparse Mode Multicast Routing Architecture

Unlike DVMRP and MOSPF, CBT [5] uses centre based shared trees: it builds and maintains a single shared bidirectional multicast distribution tree for every active multicast group in the network. This tree is rooted in a dedicated router for a multicast group that is called the *core* and it spans all group members. Here we give a short description of how a shared tree is built and how a host sends to the group.

A host starts joining a group by multicasting an IGMP[19] host membership report across its attached link. When a local CBT aware router receives this report, it invokes the tree joining process (unless it has already joined the tree) by generating a join message. This message is then sent to the next hop on the path towards the group's core router. This join message must be explicitly acknowledged either by the core router itself or by another router that is on the path between the sending router and the core, which itself has already successfully joined the tree. Once the acknowledgement reaches the router that originated the join message, a new receiver can receive the multicast traffic sent to the group. The state of the shared tree is periodically verified by exchanging of echo messages between neighbouring CBT routers on the shared tree.

Data can be sent to a CBT tree by a sender whose local router is not attached to the group tree. The sender originates native multicast data that is received by a local CBT router. This router finds out the relevant core router for the multicast group, and thus encapsulates the data packet (IP-in-IP) and unicasts it to the core router. After the core router decapsulates the packet, it disseminates the multicast data over the group shared tree. When a multicast data packet arrives at the router on the tree, the router uses the group address as an index into the multicast forwarding cache. Then, it sends a copy of the incoming multicast packet over each interface listed in the entry, except the incoming interface.

Data from the sender whose local router is already on the group tree is not sent via the core, but is distributed over the tree from the first-hop on-tree router.

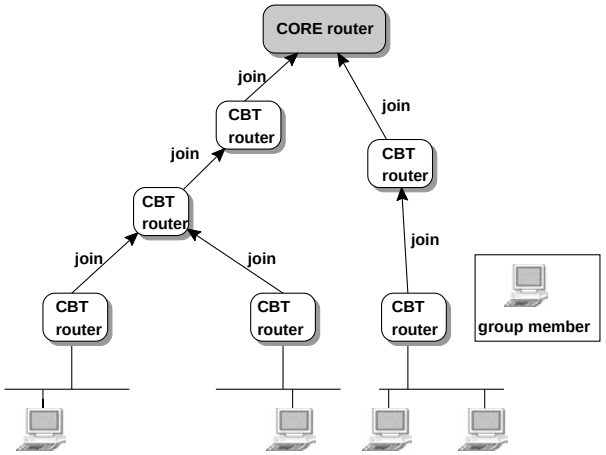


Figure 2.2: Construction of the shared tree with CBT

The main advantages of the CBT are that it is independent of the underlying unicast routing protocols and the routers keep forwarding information that correspond only to the multicast group and that do not depend on the source. This makes shared-based trees routing protocols more scalable than source-based trees routing protocols.

The main disadvantages of CBT are: CBT has a potentially higher delay compared with DVMRP because multicast packets do not take the shortest path from the source to the destinations; traffic is concentrated on the links along the shared tree; and the triangular routing problem for non-member senders.

Protocol Independent Multicast Sparse Mode (PIM-SM)

PIM-SM [17] combines the source specific shortest path trees and centre based shared trees. On one hand, PIM-SM is conceptually similar to CBT: it builds a shared directed multicast distribution tree per multicast group centered at a special router called the Rendezvous Point (RP). However, unlike CBT, PIM-SM builds unidirectional trees. The sending of multicast data is similar to CBT. Initially, the sender encapsulates data in register messages and sends them directly to the RP where the data is distributed along the shared tree. If the source continues to send, the RP may send an explicit source-specific join message towards the source. This sets up a path for traffic from the source to the RP.

On the other hand, the unique feature of PIM-SM is that for those sources whose data rate justifies it, forwarding of multicast data from a particular source to the destination group can be shifted from the shared tree onto a source-based tree. However, the result is that the routers on

the source-specific tree need to keep a source-specific state.

Multiple Centers Routing Protocols

In order to solve some of the problems inherent to both PIM-SM and CBT due to the existence of the single center router per multicast group, there are several routing protocols that introduce multiple center routers per multicast group. Hierarchical PIM (HPIM)[3] builds on PIM-SM by using the hierarchy of RPs for a group. A receiver joins the lowest level RP, this RP joins a RP at the next level and so on. The number of levels in the hierarchy depends on the scope of a multicast group. For global groups, HPIM does not perform well because all multicast data is distributed via the RP that is the highest in the hierarchy.

Multicast Source Discovery Protocol (MSDP) [18],[93],[20] allows multiple RPs per multicast group in a single share-tree PIM-SM domain. MSDP can also be used to connect several PIM-SM domains together. Members of a group initiate a sending of a join message towards the nearest RP. MSDP enables RPs that have joined members for a multicast group to learn about active sources to the group. Such RPs trigger a source specific join towards the source. Multicast data arrives at the RP along the source-tree and then is forwarded along the group shared-tree to the group members. [93] proposes to use the MSDP servers to distribute the knowledge of active multicast sources for a group. When MSDP is used to expend PIM-SM across multiple domains, MBGP [8] enables source network information to be communicated across domain boundaries.

Simple Multicast

Both CBT and PIM-SM need to discover the mapping from the group address to the corresponding core/RP. PIM-SM defines a bootstrap protocol, which all PIM-SM routers must participate in. The role of this protocol is to distribute to all PIM-SM routers the identities of candidate-RPs. Last-hop routers use a hash function to map a new group address to a candidate-RP. CBT has left the use of a bootstrap protocol as optional. Other possibility is to configure manually last-hop.

The drawback of a bootstrap protocol is that it may suffer from a slow convergence problem.

Simple Multicast (SM) [64] solves the problem of maintaining core/RP. SM is similar to CBT. SM builds a bidirectional shared tree very similar to CBT. SM offers a new multicast service model. The main novelty of SM is that it identifies a group with 8 bytes - an ordered combination (C, M) of the core IP address and multicast class D address. An end-node host conveys the pair (C, M) to its SM router. End systems may determine (C, M) for instance, via email, web advertising, or DNS lookup. This obviates the need for a bootstrap protocol between routers.

EXPRESS [30] is similar to SM [64]. EXPRESS advocates a model where a multicast tree is rooted at a single source. Receivers explicitly indicate the source when subscribing to a multicast group.

The Small Group Multicast (SGM) scheme

The SGM scheme provides the solution of supporting a very large number of small multicast groups [9]. SGM attempts to eliminate the need for multicast state maintenance in the network. SGM assumes that the source must know in advance the individual group members' IP addresses. SGM does not build the multicast tree. To deliver multicast data to group members, the source encodes the list of destinations in an SGM header and sends the packet to a router. Each SGM-aware router analyzes the SGM header, partitions the destinations based on each destination's next hop and forward an appropriate SGM packet to each of the next hops. Thus, SGM employs standard unicast IP routing, it does not need the group addressing and works the same within and across domain boundaries. Each final hop (last-hop router) removes the SGM encoding header and forwards the data to its destination as a standard unicast packet.

2.3 Architecture of DCM

In this section we describe the general concepts used by DCM. A detailed description follows in Section 2.4. We group the general concepts into three broad categories: (1) a hierarchical network model (2) how membership information is distributed and (3) how user data is forwarded.

2.3.1 Hierarchical Network Model

We consider a network model where a large single domain network is configured into areas that can be viewed as being organised in a two-level hierarchy. At the top level is a single backbone area to which all other areas connect. This is similar to what exists with OSPF[52]. In DCM we use the area concept of OSPF. However, DCM does not require underlying unicast link state routing.

Our architecture introduces several core routers per multicast group that are called Distributed Core Routers (DCRs). The DCRs are border routers situated at the edge with the backbone. Inside each non-backbone area there can exist several DCRs serving as core routers for the area.

2.3.2 Distribution of the Membership Information

Regarding the two-level hierarchical network model, we distinguish between the distribution of the membership information in non-backbone areas and in the backbone area.

Inside non-backbone areas, multicast routers keep group membership information for groups that have members inside the corresponding area. But unlike MOSPF, the group membership information is not flooded inside the area. The state information kept in multicast routers is per group ((*,G) state) and not per source per group (no (S,G) state). If for the multicast group G there are no members inside an area, then no (*,G) state is kept in that area. This is similar to MSDP when it is applied on our network model.

Inside the backbone, non-DCR routers do not keep the membership information for groups that have members in the non-backbone areas. This is different from MSDP where backbone routers can keep (S,G) information when they are on the source specific distribution trees from the senders towards RPs. This is also different from MOSPF where all backbone routers have complete knowledge of all areas' group memberships. In DCM, the backbone routers may keep group membership information for a small number of reserved multicast groups that are used for control purposes inside the backbone (as described in Section 2.4.2. We say a DCR is labeled with a multicast group when there are members of the group inside its corresponding area. DCRs in different areas run a special control protocol for distribution of the membership information, e.g information of being labeled with the multicast group.

2.3.3 Multicast Data Distribution

Multicast packets are distributed natively from the local DCR in the area to members inside the area. Multicast packets from senders inside the area are sent towards the local DCR. This can be done by encapsulation or by source routing. This is similar to MSDP.

DCRs act as packet exploders, and by using the other areas' membership information they attempt to send multicast data across the backbone only to the DCRs that need it (that are labeled with the multicast group). DCRs run a special data distribution protocol that tries to optimize the use of backbone bandwidth. The distribution trees in the backbone are source-specific, but unlike MSDP do not keep (S,G) information.

2.4 The DCM Protocol Specification

In this section we give the specification of DCM by describing the protocol mechanisms for each building block in the DCM architecture.

2.4.1 Hierarchical Network Model: Addressing Issues

In each area there are several routers that are configured to act as candidate DCRs. The identities of the candidate DCRs are known to all routers within an area by means of an intra-area bootstrap protocol [16]. This is similar to PIM-SM with the difference that the bootstrap protocol is constrained within an area. This entails a periodic distribution of the set of reachable candidate DCRs to all routers within an area.

Routers use a common hash function to map a multicast group address to one router from the set of candidate DCRs. For a particular group address *M*, we use the hash function to determine the DCR that serves[1] *M*.

The used hash function is $h(r(M), DCR_i)$. Function $r(M)$ takes as input a multicast group address and returns the range of the multicast group, while DCR_i is the unicast IP address of the DCR. The target DCR_i is then chosen as the candidate DCR with the highest value of $h(r(M), DCR_j))$ among all j from set $\{1, .., J\}$ where J is the number of candidate DCRs in an area:

$$h(r(M), DCR_i) = \max\{h(r(M), DCR_j), j = 1, .., J\} \tag{2.1}$$

One possible example of the function that gives the range[2] of the multicast group address *M* is :

$$r(M) = M\&B \text{ , where } B \text{ is a bit mask.} \tag{2.2}$$

We do not present here the hash function theory. For more information see [83], [16] and [75]. The benefits of using hashing to map a multicast group to DCR are the following:

- We achieve minimal disruption of groups when there is a change in the candidate DCR set. This means that we have to do a small number of re-mappings of multicast groups when there is a change in the candidate DCR set. See [83] for more explanations.

[1] A DCR is said to serve the multicast group address *M* when it is dynamically elected among all the candidate DCRs in the area to act as an access point for address *M*

[2] A range is the partition of the set of multicast addresses into group of addresses. A range to which a multicast group address belongs to is defined by Equation (2.2). e.g if the bit mask is (hex) 000000FF we get 256 possible ranges of IPv4 class-D addresses.

- We apply the hash function *h(.,.)* as defined by the Highest Random Weight (HRW) [75] algorithm. This function ensures load balancing between candidate DCRs. This is very important because no single DCR serves more multicast groups than any other DCR inside the same area. With this property we achieve that, when the number of candidate DCRs increases, the load on each DCR decreases. Load balancing is more efficient when the number of possible ranges of multicast addresses is larger[75].

All routers in all non-backbone areas should apply the same functions $h(.,.), r(.)$.

By applying the hash function, a candidate DCR is aware of all the ranges of multicast addresses for which it is elected to be a DCR in its area. DCRs in different areas, that serve the same range of addresses, exchange control information (see Section 2.4.2). There is one reserved multicast group that corresponds to every range of multicast addresses. In order to exchange control information with other DCRs, a DCR joins a reserved multicast group that corresponds to a range of multicast addresses that the DCR serves. Packets destined to a reserved multicast address are routed by using another multicast routing protocol, other than DCM (see Section 2.4.2). Maintaining the reserved multicast groups adds to overhead costs, and we want to keep it as low as possible. So we want to keep the number of reserved multicast groups small. Obviously there is a tradeoff between the number of reserved groups and the efficiency of load balancing among DCRs inside an area.

2.4.2 Distribution of membership information

Distribution of membership information inside non-backbone areas

When a host is interested in joining the multicast group *M*, it issues an IGMP join message. A multicast router on its LAN, known as the designated router (DR), receives the IGMP join message. The DR determines the DCR (inside its area) that serves *M* by means of a hash function, as described in the Section 2.4.1.

The process of establishing the group shared tree is similar to PIM-SM [17]. The DR sends a join message towards the determined DCR. Sending a join message forces any off-tree routers on the path to the DCR to forward a join message and join the tree. Each router on the way to the DCR keeps a forwarding state for *M*. When a join message reaches the DCR, this DCR becomes labeled with the multicast group *M*. In this way, the delivery subtree , for the receivers of the multicast group *M* in one area, is established. The subtree is maintained by periodically refreshing the state information for *M* in the routers on the subtree (this is done by periodically sending join messages).

Similar to PIM-SM, when the DR discovers that there are no longer any receivers for *M*, it sends a prune message towards the nearest DCR to disconnect from the shared distribution tree. Figure 2.3 shows an example of joining the multicast group.

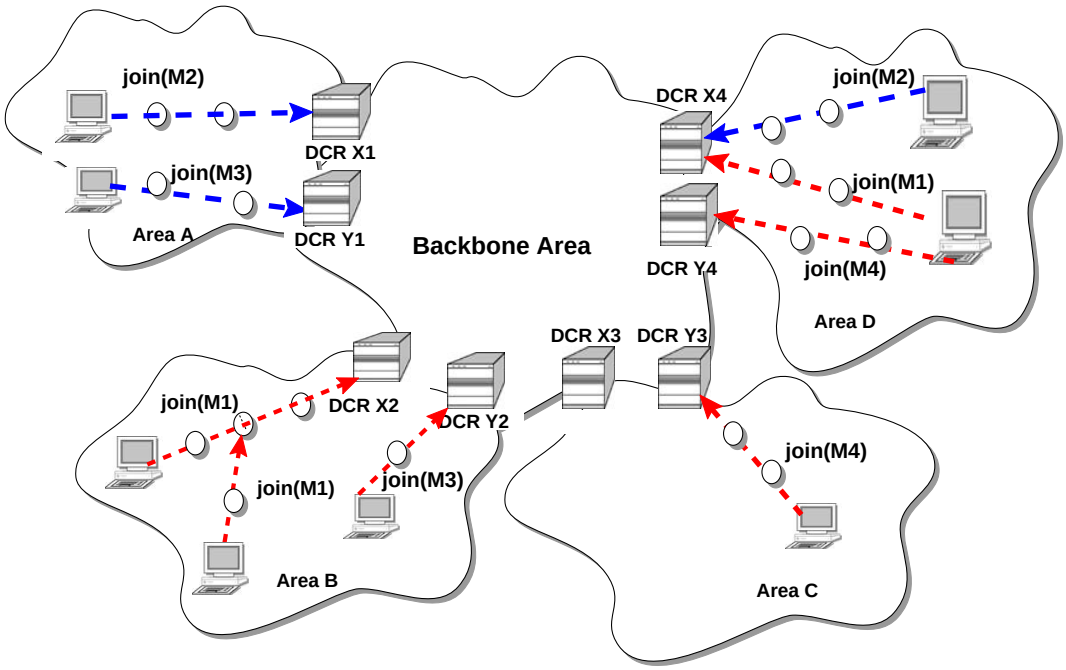


Figure 2.3: The figure shows hosts in four areas that join four multicast groups with addresses *M1*,*M2*, *M3* and *M4*. Assume that *M1* and *M2* belong to the same range of multicast addresses, while *M3* and *M4* belong to the another range of multicast addresses. Inside every area there are two DCRs. Assume that DCRs (X1,X2,X3 and X4) serve the range of multicast addresses where group addresses *M1* and *M2* belong to. DCRs (Y1,Y2,Y3 and Y4) serve the range of multicast addresses where group addresses *M3* and *M4* belong to. A circle on the figure represents a multicast router in non-backbone areas that are involved in the construction of the DCR rooted subtree. These subtrees are showed with the dash lines. X2 and X4 are now labeled with *M1*, X1 and X4 are labeled with *M2*, Y1 and Y2 are labeled with *M3*, while Y3 and Y4 are labeled with *M4*.

Distribution of membership information inside the backbone

The Membership Distribution Protocol (MDP) is used by DCRs in different areas to exchange control information. The following is the short description of MDP.

As said above, within each non-backbone area, for each range of multicast addresses (as defined by Equation (2.2)) there is one DCR serving that range. DCRs in different areas that serve the same range of multicast addresses are members of the same MDP control multicast

group that is used for exchanging control messages. This group is defined by a MDP control multicast address as described in Section 2.4.1. There are as many MDP control multicast groups as there are possible ranges of multicast addresses. A DCR joins as many MDP control multicast groups as the number of ranges of multicast addresses it serves in its area.

Each MDP multicast group has as many members as there are non-backbone areas as there is one member DCR per area. In practice, this is usually a small number. For example, in the network in Figure 2.3, X1, X2, X3 and X4 are members of the same MDP control multicast group, while Y1, Y2, Y3 and Y4 are members of the another MDP control multicast group.

We do not propose a specific protocol for maintaining the multicast tree for the MDP control multicast group. This can be done by means of an existing multicast routing protocol. For example, CBT[5] can be used.

DCRs that are members of the same MDP control multicast group exchange the following control information:

- **periodical keep-alive message.** A DCR sends periodically the keep-alive control message informing DCRs in other areas that it is alive. In this way a DCR in one area has the accurate list of DCRs in the other areas that are responsible for the same multicast groups.

- **unicast distance information.** Each DCR sends, to the corresponding MDP control multicast group, information about the unicast distance from itself to other DCRs that it has learned to serve the same range of multicast addresses. This information comes from existing unicast routing tables.

- **multicast group information** A DCR periodically notifies DCRs in other areas about multicast groups for which it is labeled. In this way, each DCR keeps a record of every other DCR that has at least one member for a multicast group from the range that the DCR serves. For example, in Figure 2.3 routers X1,X2,X3 and X4 have a list of labeled DCRs for groups *M1* and *M2,* while Y1,Y2,Y3 and Y4 keeps a list of labeled DCRs for groups *M3* and *M4*.

The next section explains how the control information exchanged with MDP is used for data distribution among DCRs.

MDP uses its MDP control multicast addresses and performs flooding inside the groups defined by those addresses. An alternative approach would be to use MDP servers. This approach leads to a more scalable, but also a more complex solution. This approach is not studied in detail in this chapter.

Comparison of DCM to MOSPF and MSDP in the backbone In MOSPF, all backbone routers have complete knowledge of all areas' group memberships. Using this information together with the backbone topology information, backbone routers calculate the multicast data distribution trees. With MOSPF, complexity in all backbone routers increases with the number of multicast groups. With DCM, DCRs are the only backbone routers that need to keep state information for the groups that they serve. In addition, as described in Section 2.4.1, the number of multicast groups that a DCR serves decreases as the number of candidate DCRs increases inside an area. Therefore, DCM is more scalable than MOSPF.

With DCM the areas' membership information is distributed among DCRs. An alternative approach, similar to what exists with MSDP, would be to distribute among DCRs the information about active sources in the domain. Then the (S,G) distribution path is built from the DCRs with the members of group G towards the source S. Under our assumptions (a large number of small multicast groups, and many senders) there would be a large number of (S,G) pairs to be maintained in backbone routers. The consequence is that backbone routers would suffer from scalability problems.

2.4.3 Multicast data distribution

How senders send to a multicast group

The sending host originates native multicast data, for the multicast group *M*, that is received by the designated router (DR) on its LAN. The DR determines the DCR within its area that serves *M*. We call this DCR the source DCR. The DR encapsulates the multicast data packet (IP-in-IP) and sends it with a destination address equal to the address of the source DCR. The source DCR receives the encapsulated multicast data.

Data distribution in the backbone

The multicast data for the group *M* is distributed from a source DCR to all DCRs that are labeled with *M*. Since we assume that the number of receivers per multicast group is not large, there are only a few labeled routers per multicast group. Our goal is to perform multicast data distribution in the backbone in such a way that backbone routers keep a minimal state information while at the same time backbone bandwidth is used efficiently. We propose a solution that can be applied in the Internet today. It uses point-to-point tunnels to perform data distribution among DCRs. With this solution, non-DCR backbone routers do not keep any state information related

to the distribution of the multicast data in the backbone. The drawback is that is that backbone bandwidth is not optimally used because using tunnels may cause possible packet duplications along backbone links. In Section 2.6 we propose two alternative solutions for data distribution in the backbone. With those solutions backbone bandwidth is used more efficiently, but at the expense of having the new routing mechanism that needs to be performed by backbone routers.

Point-to-Point Tunnels The DCR that serves the multicast group *M* keeps the following information: (1) a set *V* of DCRs that serve the same range to which *M* belongs; (2) information about unicast distances between each pair of DCRs from *V*; (3) the set *L* of labeled DCRs for *M*. The DCR obtains this information by exchanging the MDP control messages with DCRs in other areas. In this way, we present the virtual network of DCRs that serve the same range of multicast group addresses by means of an undirected complete graph $G = (V, E)$. *V* is defined above, while the set of edges *E* are tunnels between each pair of DCRs in *V*. Each edge is associated with a cost value that is equal to an inter-DCR unicast distance.

The source DCR, called *S*, calculates the optimal tree that spans the labeled DCRs. In other words, *S* finds the subtree $T = (V_T, E_T)$ of *G* that spans the set of nodes *L* such that $cost(T) = \sum_{e \in E_T} cost(e)$ is minimised. We recognise this problem as the Steiner tree problem. Instead of finding the exact solution, that is a NP-complete problem, we introduce a simple heuristic called Shortest Tunnel Heuristic (STH). STH consists of two phases. In the first phase a greedy tree is built, by adding nodes one by one, those that are closest to the tree under construction, and then removing unnecessary nodes. The second phase further improves the tree established so far.

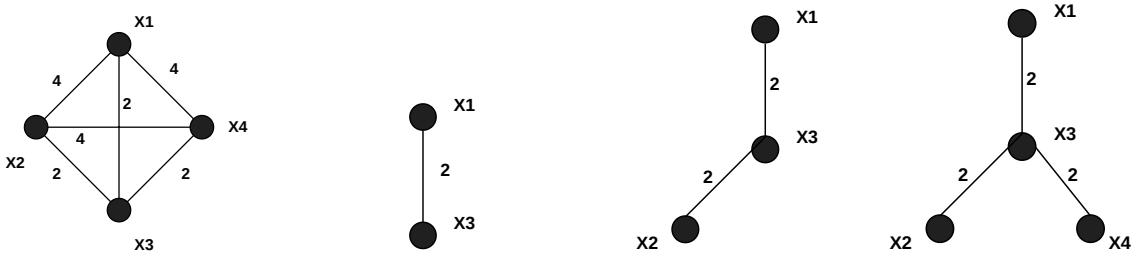
Phase 1: Build a greedy tree

- **Step 1**: Begin with a subtree *T* of *G* consisting of the singe node *S*. $k = 1$.
- **Step 2**: if $k = n$ then goto **Step 4**. n is the number of nodes in set V.
- **Step 3**: Determine node $z_{k+1} \in V$, $z_{k+1} \notin T$ closest[3] to *T* (ties are broken arbitrarily). Add node z_{k+1} to *T*. $k = k + 1$. Goto **Step 2**.
- **Step 4**: Remove from *T* non-labeled DCRs of degree[4] 1. Also remove non-labeled DCRs of degree 2 if the triangular inequality[5] holds.

[3]with the smallest cost needed to connect to some node that is already on *T*

[4]A degree of a node in a graph is the number of edges incident with a node

[5]The triangular inequality holds if the cost of a single edge that connects the nodes adjacent to the node-to-be



(a) the four node complete graph

(b) start with X1; add X3 to the tree because it is closer to X1 than X2 and X4

(c) X2 is added to the tree

(d) result of STH up to step 4; since X3 is of degree 3 it is not removed from the tree; this is STH solution since the tree cannot be improved by the steps 5 and 6.

Figure 2.4: The first example of application of STH on the complete graph

Phase 2: Improve a greedy tree

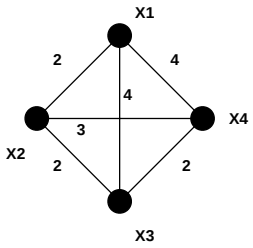
STH can be further improved by two additional steps:

- **Step 5**: Determine a minimum spanning tree for the subnetwork of *G* induced by the nodes in *T* (after the step 4).
- **Step 6**: Remove from the minimum spanning tree non-labeled DCRs of degree 1. Also remove non-labeled DCRs of degree 2 if the triangular inequality holds. The resulting tree is the (suboptimal) solution.

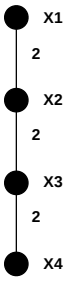
Figures 2.4, 2.5 and 2.6 illustrate three examples of the usage of STH in the network presented in Figure 2.3. Nodes X1, X2, X3 and X4 present four DCRs that serve the multicast group *M1*. In the three examples the inter-DCRs unicast distances are different. In all these examples, the source DCR for *M1* is X1 and the labeled DCRs for *M1* are X2 and X4. For the first two examples, the tree that is obtained by the first phase of STH cannot be further improved by steps 5 and 6. In the third example, steps 5 and 6 give improvements in terms of the cost of the resulting tree.

The source DCR applies STH to determine the distribution tunnel tree from itself to the list of labeled DCRs for the multicast group. The source DCR puts inter-DCR distribution information in the form of an explicit distribution list in the end-to-end option field of the packet header.

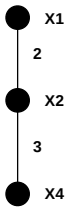
removed is less, or equal, to the sum of the costs of the two edges that connect the node-to-be removed with the two adjacent nodes. A node of degree 2 is removed by its two edges being replaced by a single edge (tunnel) connecting the two nodes adjacent to the node-to-be removed. The source DCR is never removed from a graph



(a) the four node complete graph

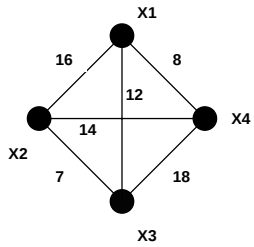


(b) result of STH up to step 4

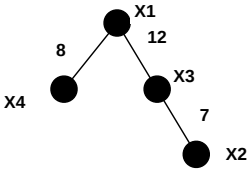


(c) X3 is of degree 2 and it is not the labeled DCR; it is removed from the tree; X2 and X4 are connected with the edge (a tunnel between them); this is STH solution since the tree cannot be improved by the steps 5 and 6.

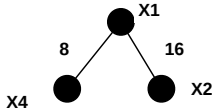
Figure 2.5: The second example of application of STH on the complete graph



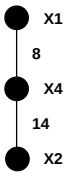
(a) the four node complete graph



(b) result of STH up to step 4



(c) X3 is removed from the tree; result of STH up to step 5



(d) Steps 5 and 6: minimum spanning tree of X1, X2 and X4; this is STH solution

Figure 2.6: The third example of application of STH on the complete graph

Under the assumption that there is a small number of receivers per multicast group, the number of labeled DCRs for a group is also small. Thus, an explicit distribution list that completely describes the distribution tunnel tree is not expected to be long.

When a DCR receives a packet from another DCR, it reads from the distribution list whether it should make a copy of the multicast data and of the identities of the DCRs where it should send multicast data by tunneling. Labeled DCRs deliver data to local receivers in the corresponding area. An example that shows how multicast data is distributed among DCRs is presented in Figure 2.7. This is a simple example when the resulting distribution tunnel tree is of height 2. Our approach works also for more complex trees, when the height of the tree is more than 2. For

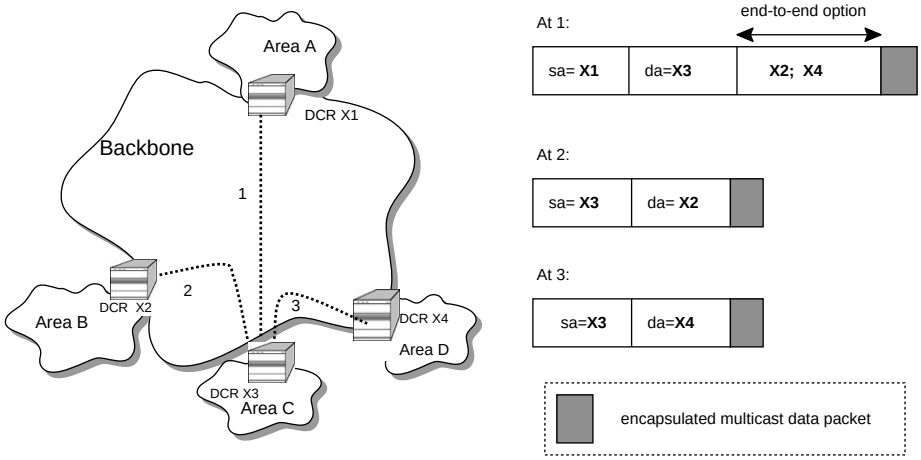


Figure 2.7: The figure presents an example of inter-DCR multicast data distribution by using point-to-point tunnels. The source DCR is X1 and labeled DCRs are X2 and X4. X1 calculates the distribution tunnel tree to X2 and X4 by applying STH. Assume that the result of STH gives the distribution tunnel tree consisting of edges X1-X3, X3-X2 and X3-X4 (as in Figure 2.4). Then X1 sends the encapsulated multicast data packet to X3. In the end-to-end option field of the packet, a distribution list is contained. X3 sends two copies of multicast data: one to X2 and the other to X4. On this figure are also presented packet formats at various points (points 1, 2 and 3) on the way from X1 to X2 and X4. A tunnel between the two DCRs is shown with the dash line.

such cases, the distribution list that describes the tree is expected to be longer (see Figure 2.11 for an example of a distribution list that describes a more complex tree).

The source DCR applies STH every time it learns that there is a change in any of the following information: list of DCRs, unicast distances between DCRs, or list of labeled DCRs for a multicast group. If the application of STH results in a new distribution tunnel tree in the backbone, subsequent data is sent along the new tree. Therefore, even though the distribution tree has changed, this does not result in data losses. Analysis of how many data packets are lost, before the source DCR learns about membership change, is to be done.

In order to minimize the encapsulation overhead while sending the multicast data in the backbone, we can use, instead of IP-in-IP tunneling, the encapsulation technique called Minimal Encapsulation within IP[58],[77]. This technique compresses the inner IP header by removing the duplicated fields that are in both inner and outer header. In this way we have less header length overhead and less MTU problems than in the case of IP-in-IP encapsulation.

Data distribution inside non-backbone area

A DCR receives encapsulated multicast data packets either from a source that is within its area, or from a DCR in another area. A DCR checks if it is labeled with the multicast group that corresponds to the received packet, i.e whether there are members of the multicast group in its area. If this is the case, a DCR forwards the multicast packet along the distribution subtree that is already established for the multicast group.

2.5 Evaluation of DCM

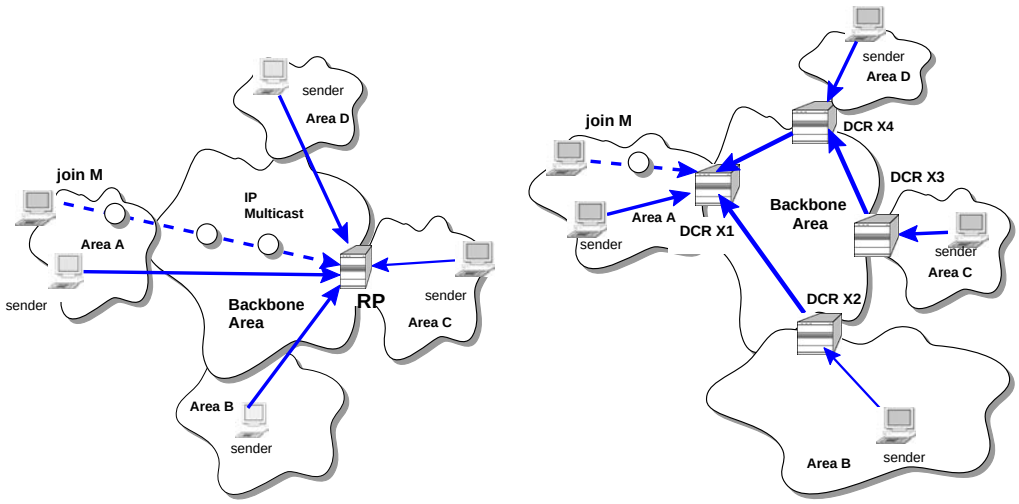
We have implemented DCM using the Network Simulator (NS) tool [1]. To examine the performance of DCM, we performed simulations on a single-domain network model consisting of four areas connected via the backbone area. Figure 2.8 illustrates the network model used in simulations where areas A,B, C and D are connected via the backbone. The whole network contains 128 nodes. We examined the performance under realistic conditions: the links on the network were configured to run at 1.5Mb/s with a 10ms delay between hops. The link costs in the backbone area are higher than the costs in other areas. We evaluate DCM and compare it with the shared-tree case of PIM-SM. Our assumptions are given in the introductory part of the chapter: there is a large number of small multicast groups and a large number of potential senders that sporadically send to a group. The most interesting example where such assumptions are satisfied is when one multicast address is assigned to a mobile host. We do not consider the case of PIM-SM when it builds source-specific trees because this introduces a high degree of complexity to PIM-SM when the number of senders is large. We analyse the following characteristics: size of the routing table, traffic concentration in the network and control traffic overhead.

We also discuss how DCM performs when there are some groups that have many members that are sparsely distributed in a large single domain network.

2.5.1 Amount of multicast router state information and CPU Usage

DCM requires that each multicast router maintains a table of multicast routing information. In our simulations, we want to check the size of multicast router routing table. This is the number of (*,G) multicast forwarding entries. The routing table size becomes an especially important issue when the number of senders and groups grows, because router speed and memory requirements are affected.

We performed several simulations. In all the simulations, we used the same network model



(a) PIM-SM Shared-tree (b) DCM

Figure 2.8: The figure presents one member of the multicast group *M* in area A and four senders in areas A, B, C and D. Two different approaches for data distribution are illustrated: the shared-tree case of PIM-SM and DCM. In the case of DCM within each area there is one DCR that serves *M*. In PIM-SM one of the DCRs is chosen to be the centre router (RP) . With PIM-SM, all senders send encapsulated multicast data to the RP. In DCM each sender sends encapsulated multicast data to the DCR inside their area. With PIM-SM, multicast data is distributed from the RP along established distribution tree to the receiver (dash line). With DCM, data is distributed from source DCRs (X1, X2, X3 and X4) to a receiver by means of point-to-point tunnels (full lines in the backbone) and the established subtree in Area A (a dash line)

presented in Figure 2.8, but with different numbers of multicast groups. For each group there are 2 receivers and 20 senders.

Within each area, there is more than one candidate DCR. The hash function is used by routers within the network to map a multicast group to one DCR in the corresponding area. We randomly distributed membership among a number of active groups. For every multicast group, receivers are chosen randomly. In the same way, senders are chosen.

The same scenarios were simulated with PIM-SM applied as the multicast routing protocol. In PIM-SM, candidate RP routers are placed at the same location as candidate DCRs in the DCM simulation.

We verified that among all routers in the network, routers with the largest routing table size are DRCs - in the case of DCM. In the case of PIM-SM, they are RPs and backbone routers. We define the most loaded router as the router with the largest routing table size. Figure 2.9 shows the routing table size in the most loaded router for the two different approaches. Figure 2.9

illustrates that the size of the routing table of the most loaded DCR is increasing linearly with the number of multicast groups. The most loaded router in PIM-SM is in the backbone. As the number of multicast groups increases, the size of the routing table in the most loaded DCR becomes considerably smaller than the size in the most loaded PIM-SM backbone router.

As it is expected, our simulation results showed that routing table size in RPs is larger than in DCRs. This can be explained by the fact that the RP router in the case of PIM-SM is responsible for the receivers and senders in the whole domain, while DCRs are responsible for receivers and senders in the area where the DCR belongs.

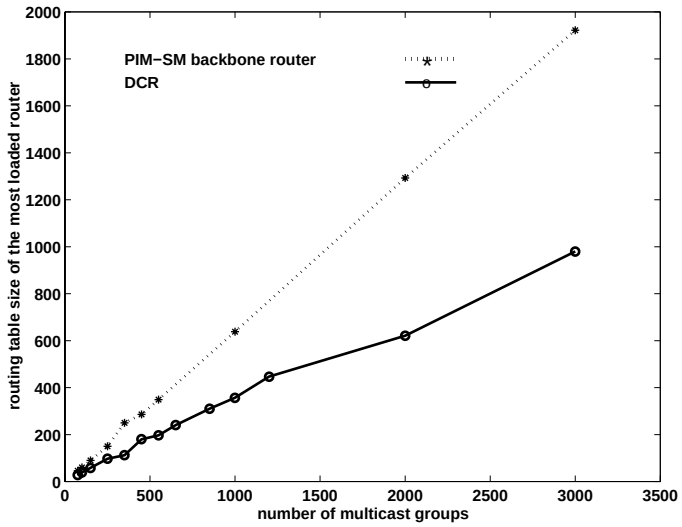


Figure 2.9: Routing table size for the most loaded routers

For non-backbone routers, simulation results show that with the placement of RPs at the edges of the backbone, there is not a big difference in their routing table sizes for DCM and PIM-SM.

Figure 2.10 illustrates the average routing table size in backbone routers for the two routing protocols. In the case of PIM-SM, this size is increasing linearly with the number of multicast groups. With DCM all join/prune messages from the receivers in non-backbone areas are terminated at the corresponding DCRs situated at the edge with the backbone. Thus in DCM, non-DCR backbone routers need not keep multicast group state information for groups with receivers inside non-backbone areas. Backbone routers may keep group membership information only for a small number of the MDP control multicast groups.

Here we also investigate how DCM compares to PIM-SM in terms of CPU. In non-backbone areas, the forwarding mechanism of multicast data in routers, other than DCRs and RPs, is the

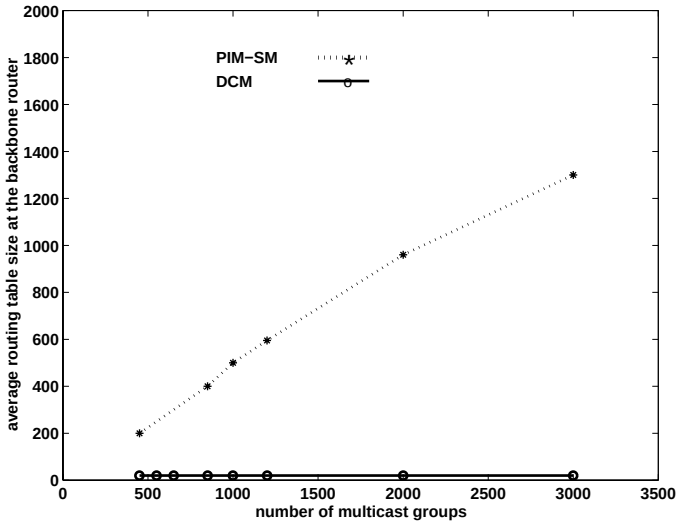


Figure 2.10: Average routing table size at backbone router

same for the two approaches. Thus, the forwarding engine in such routers costs the same in terms of CPU for the two routing protocols. However, in the case of DCM, DCRs have more complex forwarding engines than RPs in the case of PIM-SM. The reasons are that DCRs run MDP, and the special packet forwarding mechanism in the backbone. Consequently, DCRs use more CPU than RPs in the case of PIM-SM. A detailed analysis and numerical results for CPU usage for the two approaches is yet to be done.

2.5.2 Traffic concentration

In the shared-tree case of PIM-SM, every sender to a multicast group initially encapsulates data in register messages and sends them directly to the RP router uniquely assigned to the group within the whole domain. In Figure 2.8(a) all four senders of a multicast group send data towards a single point in the network. This increases traffic concentration on the links leading to the RP.

Unlike PIM-SM, CBT builds bidirectional shared trees. With CBT, data from a sender whose local router is already on the shared tree is not sent via the core, as is the case with uni-directional shared trees, but is distributed over the tree from its local on-tree router. However, in the case that we consider, when there are only a few receivers per multicast group, many senders' local routers are not on the shared tree. With CBT, in this case, data should be distributed via the core, which becomes similar to data distribution with PIM-SM.

With DCM, converging traffic is not sent to a single point in the network because each sender

sends data to the DCR assigned to a multicast group within the corresponding area (as presented in Figure 2.8(b)).

In DCM, if all senders and all receivers are in the same area, data is not forwarded to the backbone. In that way, backbone routers don't forward the local traffic generated inside an area. Thus, triangular routing across expensive backbone links is avoided.

2.5.3 Control traffic overhead

Join/prune messages are overhead messages that are used for setting up, maintaining and tearing down the multicast data delivery subtrees. In our simulations we wanted to measure the number of these messages that are exchanged in the cases when DCM and PIM-SM are used as the multicast routing protocols. Simulations are performed with the same simulation parameters as in the previous subsection (2 receivers per multicast group). They have shown that in DCM the number of join/prune messages is around 20% smaller than in PIM-SM. This result can be explained by the fact that in DCM all join/prune messages from the receivers in the non-backbone areas are terminated at the corresponding DCRs inside the same area, close to the destinations. In PIM-SM join/prune messages must reach the RP that may be far away from the destinations.

In DCM, for every MDP control multicast group, DCRs exchange the MDP control messages. As it is explained in Section 2.4.2, the number of the MDP control multicast groups is equal to the number of possible ranges of multicast group addresses. This number is set independently of the number of multicast groups in the areas. The number of members per MDP control multicast group is equal to the number of non-backbone areas. This is usually a small number. Since the senders to the MDP control multicast group are DCRs, which are the MDP control group members, the number of senders is also a small number. The overhead of the MDP keep-alive control messages depends on the time period that they are sent. DCRs also exchange the MDP control messages that notify the multicast groups for which they are labeled. Instead of sending periodically the MDP control message for every single multicast group that is serves, a DCR can send an aggregate control information for a list of multicast groups, thus reducing the MDP control traffic overhead.

2.5.4 Behaviour of DCM when the number of receivers per multicast group is not small

DCM is a sparse mode routing protocol, designed to be optimal when there are many groups with a few members. Below we investigate how DCM performs when there are some groups that have many members that are sparsely distributed in a large single domain network.

- In the case of PIM-SM, when there are more receivers per multicast group, more control join/prune messages are sent towards the RP for the multicast group. This router is probably far away from many receivers.

  In the case of DCM, join/prune messages are sent from receivers towards the nearest DCR. The number of join/prune messages in the case of DCM becomes considerably smaller than in the case of PIM-SM when the number of receivers increases. The number of the MDP control multicast groups and the MDP control traffic overhead are independent of the number of receivers per multicast group.

- DCM alleviates the triangular routing problem that is common to the shared tree case of PIM-SM. When the number of receivers increases, the triangular routing problem with PIM-SM becomes more important.

- With DCM, when the number of receivers per group increases, we can expect that there will be more labeled DCRs per group (but this number is always less that the number of areas). The time to compute the distribution tunnel tree in the backbone is equal to the time to perform STH. The required time is dependent of the number of DCRs that serve the multicast group (equal to the number of non-backbone areas) and is independent of the number of receivers per group. However, we can expect that the distribution tunnel tree in the backbone after applying STH is more complex, and that it contains more tunnel edges, since the number of labeled routers is larger and more nodes need to be spanned.

2.6 Data Distribution in the backbone - two approaches without tunneling

With DCM, data distribution in the backbone uses point-to-point tunnels between DCRs. With this approach backbone routers other than DCRs need not be multicast able, but the consequence is that it does not completely optimize the use of backbone bandwidth. In order to make the

data distribution more optimal, backbone routers should also be included in the forwarding of multicast data.

Here we give the overview of two alternative solutions called tree-based source routing and list-based source routing that use backbone bandwidth more optimal than point-to-point approach. The complete specification of these solutions is the subject of our future work.

Tree-Based Source Routing

This solution assumes that the DCRs are aware of backbone topology (e.g the backbone is one OSPF area) and backbone routers implement a special packet forwarding mechanism called tree source routing. This approach consists in that a source DCR for the multicast group computes a shortest path tree rooted at itself to a list of labeled DCRs for the multicast group. DCRs in other areas that serve the multicast address, as well as non-DCR backbone routers, can be included in a shortest path. A description of a shortest path tree with destinations and branching points is included in the tree source routing header by the source DCR. Figure 2.11 shows one example of the tree source routing approach.

This approach ensures that backbone bandwidth is used more optimally than if the "point-to-point tunnels" approach is used. This is achieved at the expense of introducing the new tree source routing mechanism that needs to be performed by backbone routers.

List-Based Source Routing

This solution proposes a new list-based multicast data distribution in the backbone. Here we give an initial description of this mechanism. The final solution is the object of future research.

As in the previous approach we assume that the DCRs are aware of the backbone topology. A special list-based source routing protocol is performed by the DCRs and backbone routers. This works as follows: as soon as a source DCR determines that it must forward a packet to a list of DCRs, it determines the next backbone router(s) to which it should send a copy of the packet to reach every listed DCR. The source DCR sends a copy of the packet to each determined router together with a sublist of the DCRs that should be reached from this router. This sublist is contained in a list source routing header. This is similar to the IP option field that is described in [26]. Unlike a tree-based source routing header, where non-DCR backbone routers can be also included in a tree source routing header, the list source routing header contains only the final DCR destinations.

Each backbone router performs the same steps until multicast data has reached every labeled

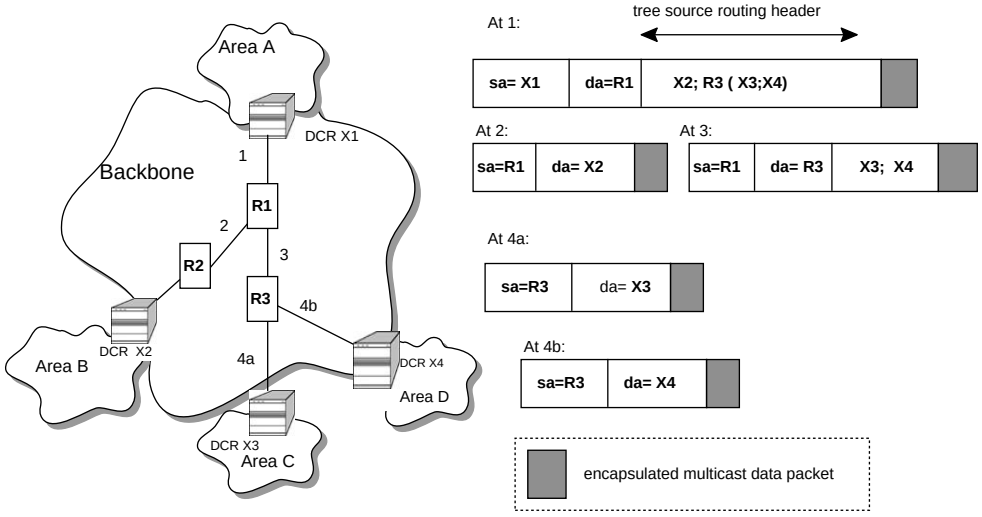


Figure 2.11: This figure shows how multicast data is distributed from source DCR X1 to labeled DCRs X2, X3 and X4 by using tree-based source routing approach. X1 puts distribution information in tree source routing header after computing a shortest-path tree to routers X2, X3 and X4. At first, the data should be delivered to backbone router R1 where two copies of the multicast data are made. One copy is sent encapsulated to X2, while the other is sent encapsulated to backbone router R3. As soon as router R3 receives a packet it reads from the tree source routing header that it should send two copies of the multicast data: one to X3 and the other to X4.

DCR. Note that a DCR can also send a copy directly to another DCR.

SGM [9] uses a similar approach to the list-based source routing method to deliver multicast data. However, in SGM the list contains individual group members' IP addresses.

Figure 2.12 presents one example of list-based source routing approach.

2.7 Examples of the application of DCM

2.7.1 An example of the application of DCM in distributed simulations

Distributed simulations and distributed games are applications where scalable multicast communication is needed to support a large number of participants. [49] describes a network architecture for solving the problem of scaling very large distributed simulations. A large-scale virtual environment is spatially partitioned into appropriately sized hexagonal cells. Each cell is mapped to a multicast group. For a large virtual environment there are a large number of multicast groups. Each participant is associated with a number of cells according to its area of interest, and it joins

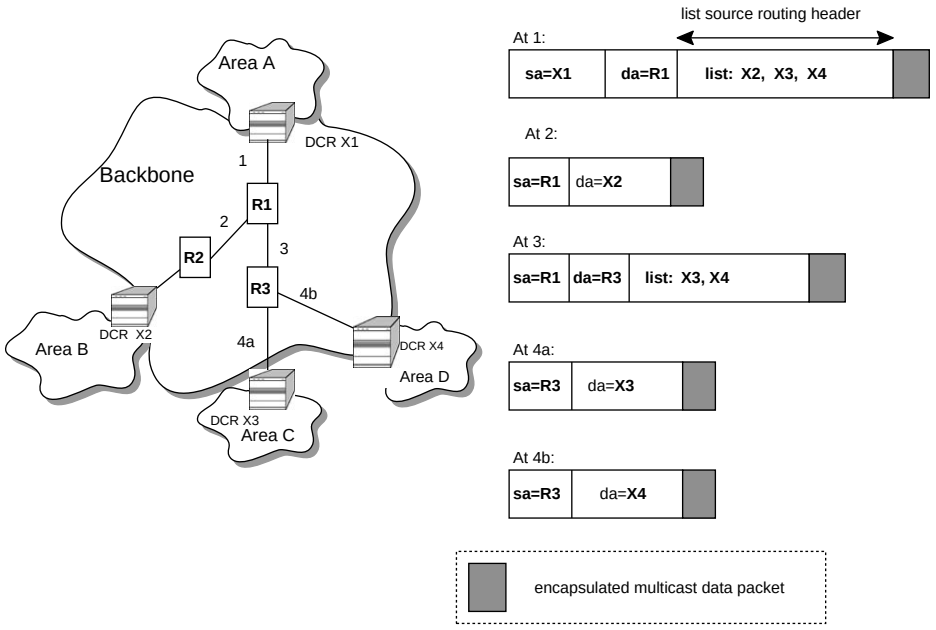


Figure 2.12: This figure shows how multicast data is distributed from the source DCR X1 to labeled DCRs X2, X3 and X4 by using the list-based source routing approach. X1 determines that it should send a copy of multicast data to backbone router R1. Router X1 puts in the list source routing header information that X2, X3 and X4 should be reached from R1. As soon as R1 receives a packet it makes two copies of the multicast data. One copy of the multicast data is encapsulated is a packet that is sent to X2. Another copy of the multicast data is sent to R3. This packet contains in the list source routing header a list of DCRs that should be reached from R3. The list contains X3 and X4. As soon as R3 receives a packet from R1 it makes two copies of the multicast data. One copy is sent encapsulated to X3. Another copy is sent encapsulated to X4.

the corresponding multicast groups. Each participant have the view of all other participants that are members of the same multicast group. Participants can move and dynamically change their cells of interest. We can assume that in a large virtual environment the number of participants per cell is not a large number. In this case DCM can be applied to route packets to a multicast group inside the cell.

2.7.2 Example of application of DCM: supporting host mobility

Another application of DCM is the use of it to route packets to the mobile hosts. We start this subsection with a short description of certain existing proposals for providing host mobility in the Internet and then illustrate how DCM can support mobility.

Overview of proposals for providing host mobility in the Internet

In the IETF Mobile IP proposal [57] each host has a permanent *home IP* address that does not change regardless of the mobile host's current location. When the mobile host visits a foreign network, it is associated with a care-of-address, that is an IP address related with the mobile host current position in the Internet. When a host moves to a visited network it registers its new location with its home agent. The home agent is a machine that acts as a proxy on behalf of the mobile host when it is absent. When some stationary host sends packets for the mobile host it addresses them to the mobile host's home address. When packets arrive on the mobile host's home network, the home agent intercepts them and sends, by encapsulation, packets towards the mobile host's current location. With this approach all datagrams addressed to a mobile host are always routed via its home agent. This causes the so-called triangle routing problem.

In IPv6 mobility proposal[61] when a handover is performed, the mobile host is responsible for informing its home agent and correspondent hosts about its new location. In order to reduce packet losses during handover, [61] proposes a router-assisted smooth handover.

The Columbia approach [32] was designed to support intracampus mobility. Each mobile host always retains one IP home address, regardless of where it is on the network. There is a number of dedicated Mobile Support Stations (MSSs) that are used to assure the mobile host's reachability. Each mobile host is always reachable via one of the MSSs. When a mobile host changes its location it has to register with a new MSS. A MSS is thus aware of all registered mobile hosts in its wireless cell. A source that wants to send a packet to a mobile host sends it to the MSS that is closest to the source host. This MSS is responsible for learning about the MSS that is closest to the mobile host and to deliver the packet. A special protocol is used to exchange information among MSSs.

Seshan et al. [69] propose a scheme in which each mobile host is assigned a temporary IP multicast address by its home agent. Then a home agent encapsulates packets destined for the mobile host and forwards them to its assigned multicast group. While only one base station actively forwards packets to the mobile host, nearby base stations are asked by the mobile host to join the multicast group. These stations buffer the recent packets and can quickly forward them to the mobile host should a handover happen.

MSM-IP (Mobility support using Multicasting in IP) [54] proposes a generic architecture to support host mobility in the Internet by using multicasting as a mechanism to route packets to the mobile hosts. The routing protocol used in this architecture is out of the scope of MSM-IP. Both papers [69] and [54] inform that the existing multicast routing protocols do not perform well in order to support host mobility.

Cellular IP [82] architecture relies on the separation of local mobility from wide area mobility. Cellular IP is applied in a wireless access network and it can interwork with Mobile IP to provide wide area mobility support, that is mobility between Cellular IP networks. With the Cellular IP network, nodes maintain distributed caches for location management and routing purposes. Distributed paging cache coarsely maintains the position of 'idle' mobile hosts in a cellular IP network. Distributed routing cache maintains the position of active mobile hosts in a Cellular IP network and is updated more frequently that a paging cache. In a Cellular IP network there exists one gateway node (GW). A mobile host entering a Cellular IP network communicates the local GW's address to its home agent as care-of-address. All packets for the mobile host enter a Cellular IP network via the GW. From the GW, packets addressed to a mobile host are routed to its current base station on a hop-by-hop basis according to routing caches in the network nodes.

Application of DCM to host mobility

In this section we show how DCM can be applied in the mobility management approach based on multicasting. This approach is not an alternative to Mobile IP [57] because DCM is not a solution to wide-area mobility. In contrast, this approach can be used as an alternative to Cellular IP[82] within a single domain network.

We consider the network environment composed of wireless cells. Mobile hosts communicate with base stations over wireless links, while the base stations have a fixed connections to the Internet.

When a visiting mobile host arrives in the new domain it is assigned a temporary multicast address[6]. This is the care-of address that the mobile keeps as long as it stays in the same domain. This is unlike Mobile IP [57] where the mobile host does a location update after each migration and informs to its possible distant home agent.

We propose to use DCM as the mechanism to route packets to the mobile hosts. As explained in Section 2.4.1, for the mobile host's assigned multicast address, within each area, there is a DCR that serves that multicast address. These DCRs are responsible for forwarding packets to the mobile host. As said before, the DCRs run the MDP control protocol and are members of a MDP control multicast group for exchanging MDP control information.

A multicast router in the mobile host's cell initiates a joining the multicast group assigned to the mobile host. Typically this router coexists with the base station in the cell. As described in Section 2.4.3, the join message is propagated to the DCR inside the area that serves the mobile

[6]In this chapter we do not discuss how a multicast address is assigned to a mobile host

host's multicast address. Then, the DCR sends to the MDP control multicast group a MDP control message when the mobile host is registered.

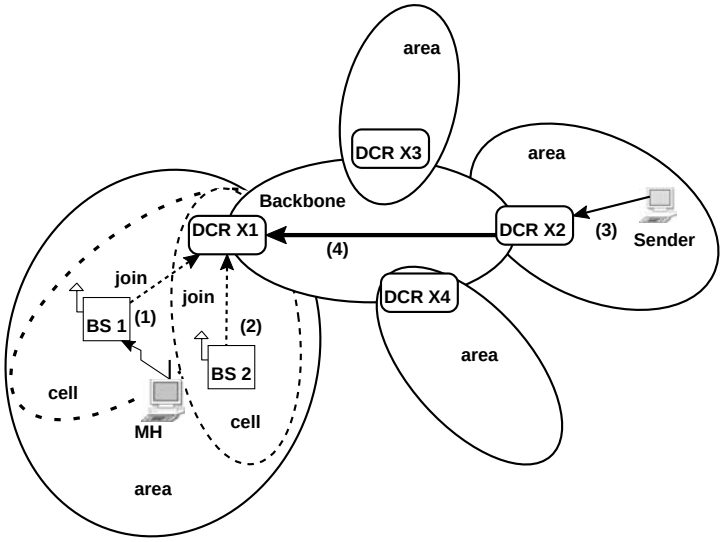


Figure 2.13: Illustration of the multicast-based mobility management. The mobile host (MH) is assigned the multicast address *M*. Four DCRs, X1, X2, X3 and X4 serve *M*. Step (1): Base station BS1 sends a join message for *M* towards X1. X1 informs X2, X3 and X4 that it has a member for *M*. Step (2): Advance registration for *M* in a neighbouring cell is done by BS2. Step (3): The sender sends a packet to multicast group *M*. Step (4): The packet gets delivered through the backbone to X1. Step (5): X1 receives encapsulated multicast data packet. From X1 data is forwarded to BS1 and BS2. MH receives data from BS1.

In order to reduce packet latency and losses during a handover, advance registration can be performed. When a mobile host moves to a new cell, the base station in the new cell should already start receiving data for the mobile host. The mobile host continues to receive the data without disruption. There are several ways to perform this:

- A base station that anticipates[7] the arrival of a mobile host initiates joining the multicast address assigned to the mobile host. This is illustrated in one example in Figure 2.13.
- In the case where a bandwidth can be afforded on the wired network, all neighbouring base stations can start receiving data destined to a mobile host. This guarantees that there would be no latency and packet losses during a handover.

A packet for the mobile host reaches all base stations that joined the multicast group assigned to the mobile host. At the same time the mobile host receives data only from a base station in its

[7]The mechanism by which the base station anticipates the arrival of the mobile host is out of the scope of our work

current cell. A base station that receives a packet on behalf of the mobile host that is not present in its cell can either discard a packet or buffer it for a certain interval of time (e.g. 10ms). Further research is needed to determine what is the best approach.

Here we describe in more detail how advance registration is performed. At its current cell, the mobile host receives data along the distribution subtree that is established for the mobile host's multicast address. This tree is rooted at the DCR and maintained with a periodical sending of the join messages. Now, suppose that the base station in the neighbouring cell anticipates the arrival of the mobile host. It begins a joining process for the multicast group assigned to the mobile host. This process is terminated when a join message reaches a router that is already on the distribution tree. When the cells are close to each other, joining is terminated at the lowest branching point in the distribution tree. This ensures that the neighbouring base station quickly becomes a part of the multicast distribution tree with low overhead. The neighbouring base station can start joining the multicast group assigned to the mobile host after the mobile host leaves its previous cell. Routers on the distribution tree keep forwarding information for a given time, even if the previous base station stops refreshing the tree because the mobile host leaves its cell. As before, if the base stations are close to each other, the multicast distribution tree for the new base station can be established in a short period of time thus making handover efficient. One example that illustrates advance registration is presented in Figure 2.14.

Comparison of the mobility management based on DCM and the Cellular IP approach

In Cellular IP, the process of establishing the distribution tree from the gateway node to the mobile host is similar to what exists in DCM for establishing the distribution subtree from a DCR to the mobile host in its current area. With DCM, maintenance of the distribution tree is performed by the sending of periodic join messages and is initiated by the base stations in the vicinity of the mobile host. With Cellular IP, this is done on the packet basis sent from the active mobile.

We see the scalability problem with Cellular IP when there is a large number of mobile hosts inside the Cellular IP network. The single gateway node is the centre of all distribution trees that are built for mobile hosts within a network. All the traffic for mobile hosts inside the Cellular IP networks goes via the gateway node that presents a 'hot spot' in the network.

With DCM we avoid existence of the center router, a potential 'host spot' in the network. DCM builds distribution subtrees for mobile hosts that are rooted at a number of DCRs. We

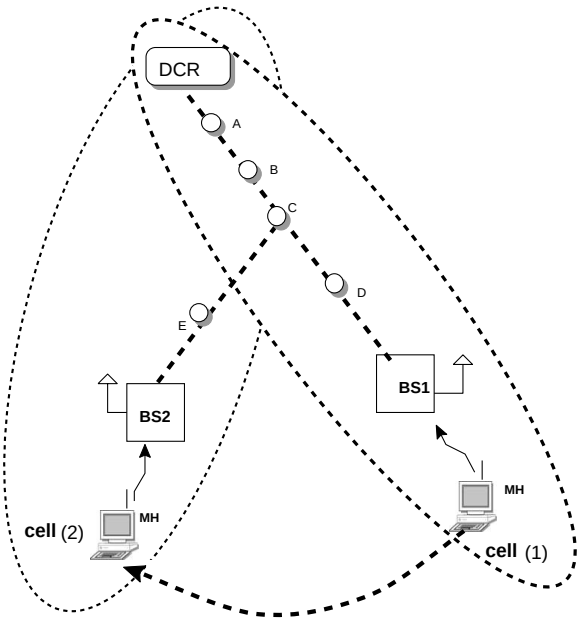


Figure 2.14: This figure presents an example of advance registration. At first, the mobile host (MH) is in cell 1. MH is assigned a multicast address *M*. Base station BS1 receives data for MH along the distribution subtree rooted at the DCR. On this subtree are routers A, B, C and D. Before a host moves from cell 1 to cell 2, neighbouring base station BS2 initiates an advance joining for *M*. Joining at position 2 is terminated at router C.

believe DCM scales better than Cellular IP when there is a large number of mobile hosts.

2.8 DCM and cellular Internet Telephony

In this section we show how DCM can be used to route packets to mobile hosts in cellular Internet telephony (IPtel). We consider the network environment composed of wireless cells. Mobile hosts communicate with base stations over wireless links. We assume that Session Initiation Protocol (SIP)[27] is used to establish, modify and terminate IPtel calls. Here we describe how DCM can be used in conjunction with SIP to support terminal mobility. Terminal mobility is the ability to maintain a communication when a host is moving from one location to another during the call.

The owner of the mobile host is identified by its SIP URL address. A SIP server in the mobile host's home domain can be identified from this address. When the mobile host moves into the new domain it is assigned a temporary multicast address that it keeps as long as it stays in the same domain (how a multicast address is assigned to a mobile host is out of the scope

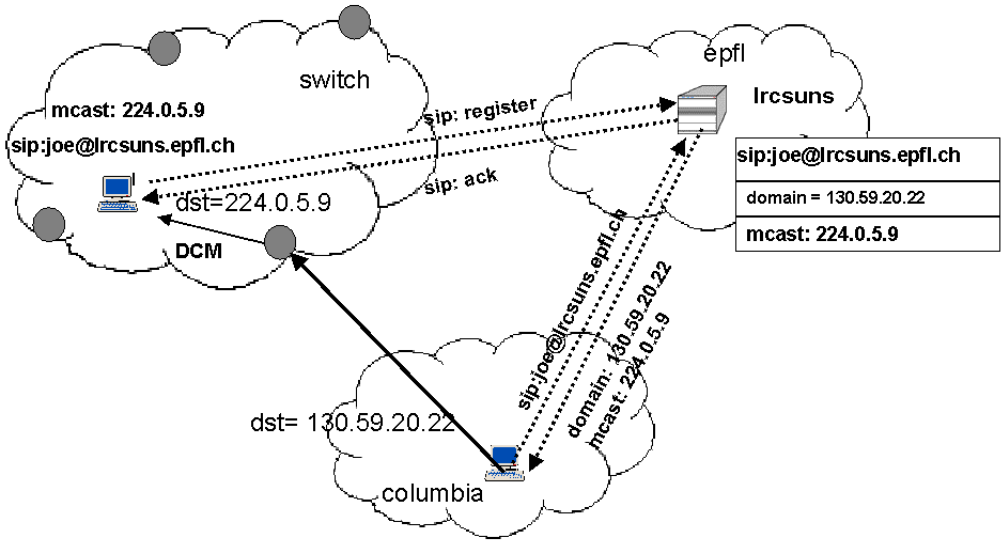


Figure 2.15: Figure illustrates how DCM can be used in conjunction with the SIP protocol to support terminal mobility in IP telephony. First, the mobile host (joe) that is now in the foreign domain ($switch$) registers with a SIP server in its home domain ($epfl$): the mobile informs about its current domain's anycast address (130.59.20.22) and the assigned multicast address (224.0.5.9). Second, a caller (in $columbia$ domain) that wants to establish the call with the mobile host, contacts the mobile host's home domain SIP server and learns the mobile host's current domain anycast address and multicast address. Third, in the case of IPv4, a caller sends multicast packets for the mobile host encapsulated to the anycast address of the mobile host's current domain. Then, the border router decapsulates the multicast packet, and a packet is forwarded to the mobile host by using DCM.

of this thesis). Then the mobile registers with a SIP server in its home domain in order to be found. During the registration process, the mobile host sends to its home SIP server the anycast address of its current domain and its assigned multicast address. In each domain there are border routers that are configured with the domain anycast address and are responsible for accepting and forwarding packets for the mobile hosts that are in the domain. The domain anycast address is a reserved unicast address. Border routers configured with the anycast address recognise the anycast address as one of their logical interfaces. The routing of packets to the anycast address is done by standard unicast routing mechanisms.

A caller that wants to establish a call with the mobile host know of the mobile host's SIP URL address. Then a caller contacts a SIP server in the mobile host's home domain for the mobile host's current location. A SIP server acts in redirect mode and returns to the caller information about the anycast address of the mobile host's current domain and its assigned multicast address. A caller sends to the mobile host a SIP INVITE message. If the caller is also mobile, it informs the callee about its domain's anycast address and its assigned multicast address.

When the sender to the mobile host and the mobile host are in the same domain, packets for the mobile host are destined to the mobile host's multicast address and are routed by DCM. This is explained in Section 2.4.

If the sender and the mobile host are in different domains, multicast packets to the mobile host should first reach the domain where the mobile lies. We distinguish two cases to achieve this depending on the used version of the IP protocol:

- In the case of IPv6, a source sends a packet to the mobile host by using a loose source routing option. A source sets the destination field of the packet header to the multicast address assigned to the mobile host and the IPv6 routing header is set to the mobile host's domain router anycast address. A packet is routed to the nearest border router in the mobile host's domain that is configured with the anycast address. The next address to be visited is the multicast address assigned to the mobile host. DCM is used to route packets from the border router to the mobile host.

- In the case of IPv4, the sender sends multicast packets for the mobile host encapsulated (IP-in-IP) to the anycast address of the mobile host's current domain. The nearest border router that is configured with the anycast address decapsulates the multicast packet and, as in the case of IPv6, a packet is forwarded to the mobile host by using DCM.

As explained in Section 2.3, for the mobile host's assigned multicast address, within each area, there is a DCR that serves that multicast address. Those DCRs are responsible for forward-

ing packets to the mobile host. As said before, the DCRs run the MDP control protocol and are members of a MDP control multicast group for exchanging MDP control information.

A multicast router in the mobile host's cell initiates a joining the multicast group assigned to the mobile host. Typically this router coexists with the base station in the cell. As described in Section 2.4.2 the join message is propagated to the DCR inside the area that serves the mobile host's multicast address. Then, the DCR sends to the MDP control multicast group a MDP control message when the mobile host is registered.

Figure 2.15 illustrates with one example how DCM can be used in conjunction with the SIP protocol to support terminal mobility in IP telephony.

Note

In this chapter we do not address the problems of using multicast routing to support end-to-end unicast communication. These problems are related to protocols such as: TCP, ICMP, IGMP, ARP. A simple solution to this problem is to have a special range of unicast addresses that are routed as multicast addresses. In this way, packets destined to the mobile host are routed by using a multicast mechanism. Conversely, at the end systems, these packets are considered as unicast packets and standard unicast mechanisms are applied.

2.9 Conclusion

We have considered the problem of multicast routing in a large single domain network with a very large number of multicast groups with a small number of receivers. Our proposal, called Distributed Core Multicast (DCM) is based on an extension of the centre-based tree approach. DCM uses several core routers, called Distributed Core Routers (DCRs) and a special control protocol among them. The objectives achieved with DCM are: (1) avoiding state information in backbone routers, (2) avoiding triangular routing across expensive backbone links, (3) scaling well with the number of multicast groups. Our results indicated that DCM performs better than the existing sparse mode routing protocols in terms of multicast forwarding table size. We have presented an example of the application of DCM where it is used to support host mobility in a large Internet single domain network.

Chapter 3

Routing Protocols for Mobile Ad-Hoc Networks

3.1 Introduction

In this chapter we give overview of existing routing protocols for mobile ad hoc networks. In Section 3.2 we give the overview of traditional routing protocols that do not use information about nodes positions. In Section 3.3, we give the overview of position-based routing protocols for mobile ad hoc networks. These protocols use geographic forwarding in order to scale better than traditional protocols. We describe in more details this second class of routing protocols because our solution for routing in mobile ad hoc network (described in Chapters 4, 5) falls into the position-based class. We identify the problem with existing position-based routing protocols, and how this impacts our work.

A mobile ad hoc network can be envisioned as a collection of nodes which are free to move arbitrarily. The mobility of the nodes and the variability of other connecting factors result in a network with potentially rapid and unpredictable changing topology. Ad hoc networks, may or may not be connected with the infrastructure such as Internet, but still are available for use by a group of wireless mobile hosts that operates without any base station or any centralized control.

Recently, there is a large interest for studying mobile ad hoc networks. Networks using ad hoc configuration concepts can be used in many applications. Some of the typical applications of ad hoc networks are in scenarios where it is difficult to set up a communication infrastructure either because of mobility, or because this is expensive. Examples of such applications are military applications and rescue operations, as well as for instant conferencing in infrastructure-absent geoghraphic areas. Sensor networks [40] is another area of use of mobile ad hoc networks,

currently being researched.

A mobile ad hoc networking (MANET) working group [50] within the Internet Engineering Task Force (IETF) aim to develop a routing framework for IP-based protocols in ad hoc networks. Many routing protocols have been proposed for consideration of standardization [12, 59, 34, 62, 56, 33, 22]. Most of the currently available solutions are not designed to scale to more that few hundreds nodes.

3.2 Routing protocols that do not use position information - State of the Art

Existing routing protocols for ad hoc networks fall into two broad categories. Those are proactive and reactive routing protocols. In proactive protocols very node proactively maintains routes to other nodes, so that the route is already available when it is needed for a packet to be forwarded. Reactive (on-demand) routing protocols track routes for source-destination pair that are currently communicating.

Most of the routing protocols found in the literature are variants of proactive of reactive approach, some of them combining the two. Hybrid routing protocols combine the proactive and reactive methods in one protocol. Cluster-based family of routing protocols organize the network in clusters and perform routing using the clustered network.

In the following, we briefly describe the representative protocols of each of these families of routing protocols.

3.2.1 Proactive protocols

In proactive routing protocols neighbouring nodes exchange route information periodically or each time a change occurs in the network topology. This has the advantage of minimizing delay in obtaining the a route when initiating communication to a destination. The drawback is when the mobility rate in the network is high, proactive protocols consume a large amount of network bandwith for tracking routes that may not be used before a topology change.

Destination Sequenced Distance Vector(DSDV)[60] is a variant of traditional distance-vector algorithms. It avoids loops by tagging route information for each destination with a sequence number originated by the destination. It also prevents routing fluctuations by delaying advertisements of possibly unstable routes.

Optimized Link State Routing(OLSR)[34] is a variant of the link state algorithm designed

to scale better in the environment with the frequently changing topology. In OLSR, each node includes only a subset of its neighbours in a link-state updates. OLSR introduces the concept of a multi-point relay. A node's multi-point relay is a minimal subset of its one-hop neighbours which must rebroadcast a message so that it is received by all its two-hop neighbours. Multi-point relays help in two ways. First, nodes only broadcasts the state of nodes in its multi-point relay set and thus reduce the size of link-state messages. And, second, usage of the multi-point relays minimizes flooding of control traffic (link-state updates) by using only the selected nodes.

Fisheye State Routing (FSR) [33] is a link state type routing protocol where every node maintains a topology map at each node. FSR introduces the following modifications in order to reduce the routing overhead. First, link states are not flooded. Instead, only neighbouring nodes exchange the link state information. Second, link state advertisements are only time triggered, but not event triggered. Third, instead of transmitting the entire link state information at each time when the link state information is changed, a node exchanges with its neighbours more frequently entries that correspond to nodes that are nearby (within a $scope$, that is predefined). Entries that correspond to nodes that are out of the predefined scope are exchanged less frequently. With these modifications, FSR succeeds to reduce the size of control packets and the frequency of their transmissions. FSR is functionally similar to link-state in that it maintains a network topology map at each node. However, FSR does not try to keep precise knowledge of the best path to all nodes in a network. Imprecise knowledge of the best path to a distant destination is compensated by the fact that the route becomes progressively more accurate as the packet gets closer to the destination. Thus, FSR scales better to large networks than classical link state routing protocols. However, FSR may not perform well in the case of high mobility. The reasons are that the update messages in FSR are only time-triggered and route to remote destinations become less accurate as mobility increases. As a result, some of the link state information maintained in route tables is imprecise.

Landmark Ad hoc Routing Protocol (LANMAR)[22] is a proactive routing protocol that combines characteristics of link state and distance vector routing protocols. LANMAR has a notion of landmarks to keep track of logical subnets. The concept of landmark routing was first introduced in wired area networks [79]. In LANMAR, a subnet consists of members which have a common interests and are likely to move as a "group". A "landmark" is selected in each subnet. The routing scheme is modified version of FSR [33]. The main difference is that the FSR routing table consists all nodes in the network, while the LANMAR routing table includes only the nodes within a scope and the landmark nodes. Every node proactively maintains routes to other nodes within its scope (using FSR, or the modified DSDV [60] in which the hop distance can be used

to bind the scope for routing message updating). An intermediate node that has a packet to relay first looks in its routing table to check whether the destination is within its scope. If this is the case, the packet is forwarded to the next hop towards the destination, as from the routing table. If this is not the case, the logical subnet of the destination is searched, and the packet is routed towards the landmark for that logical subnet. The packet however does not have to pass via the landmark of that logical subnet. Rather, once the packet gets to some node close to the destination and that finds the destination within its scope, the packet is forwarded directly to the destination. [22] reports that LANMAR performs better than FSR in the case of high mobility. LANMAR keeps accurate routes to all landmark nodes, unlike FSR that keeps inaccurate routes to all nodes. [22] demonstrates good performance of LANMAR under very high traffic load.

The Wireless Routing Protocol WRP [53] modifies the traditional distance-vector routing protocol in order to eliminate the counting-to-infinity problem, and reduce the occurrence of temporal loops, often with less control traffic than traditional distance-vector schemes.

3.2.2 On-demand (reactive) routing protocols

An attempt to overcome limitations of the proactive routing protocols, is to look for a route only on demand. This is the basic idea of on-demand routing protocols. In reactive protocols a control message is sent to discover a route to a given destination. Reactive protocols have smaller control traffic overhead than proactive protocols. However, since a route has to be discovered before the actual transmission of the data, these protocols can have a longer delay. Further more, due to mobility, the discovered route may be unusable since some links of the route may be broken.

Below we give the overview of the on-demand routing protocols.

In DSR [12], when a source S needs a route to a destination node D, S first checks if some of its neighbours possesses the route in question. If this is not the case, S floods the network with a *route request* for the destination node. When the request reaches the destination, the destination returns a *route reply* to the request's originator. The reply message contains the list of all intermediate nodes from S to D. Then S uses source routing with the acquired source route to send packets to D. Several methods are proposed for limiting the propagation of requests. One of these is that nodes cache the route that they learn or overhear, so that intermediate nodes can reply on behalf of the destination if the route to the destination is known. One node can cache several source routes per destination, and use them in the case if the current route breaks. If any link on a source route is broken, the source node is notified using a *route error* packet. Then the source initiates a new source route discovery to learn the valid route to the destination.

AODV [59], similarly to DSR, discovers routes on demand when the routes are needed.

However, the way how AODV maintains routing information is different from DSR. AODV uses traditional routing tables, one entry per destination. Similarly to DSR, route discovery works by flooding the network with *route request* packets. This will set entries in the nodes to propagate the route *reply packet* packet from the destination back to the source, and subsequently, to route data packets to the destination. AODV uses sequence numbers maintained at each destination to prevent routing loops. Every entry in the routing table is associated with a timer-based state. If the entry is not recently used it expires. All predecessor nodes that used this entry are notified with a *route error* packet. This notification is propagated in the network, such that eventually all routes that used this link are erased.

TORA[56] is a reactive protocol based on the earlier "link reversal" algorithms. Based on query/reply flooding process, a sequence of directed links leading from the source to the destination is formed. TORA builds a destination-oriented directed acyclic graph (DAG) for each destination. The DAG is self-adapting to the topological changes in the network. TORA provides multiple paths to a destination and ensures that they are loop-free. TORA is able to detect network partitions. Once the DAG is created, new links are not taken into consideration, unless the DAG becomes disconnected. Therefore, the route may become non-optimal. In addition, TORA suffers from high routing overhead since is bases route discovery on flooding. In addition, TORA requires reliable, ordered broadcast in order to prevent long-lived routing loops.

3.2.3 Hybrid routing protocols

In order to overcome mentioned problems of basic proactive and reactive routing protocols, hybrid routing protocols combine both a proactive and reactive approach.

ZRP[62] is one hybrid protocol. In ZRP, every node proactively maintains routes to other nodes whose distance is less than a certain number of hops (its zone). Within a zone, the proactive intra-zone routing protocol (IARP) is used. IARP can be either a distance-vector or a link-state proactive routing scheme, modified such that each node maintains the proactive routing information only within its zone. When the source does not find the destination within its zone, it invokes a reactive inter-zone routing protocol (IERP). Unlike other reactive protocols (DSR, AODV), IERP does not flood the network to discover the route. It uses the method that is called $bordercast$. This method consists in the following: the source broadcasts a *route request* to border nodes of its zone. On receiving the request, each border node verifies if the destination is within its zone. If this is not the case, the border node adds its identity in the request and re-broadcasts the request to its border nodes. When the request reaches a border node that finds the destination within its zone, the accumulated list of border nodes is returned to the source.

This list is used by the source as a loose source route to the destination. ZRP proposes several methods to optimize the route discovery algorithm. This includes using of the IARP topology information to prevent backward search and selective bordercasting. However, route discovery in the case of dense traffic patters and highly mobile network can result in high routing load and latency problem.

3.2.4 Cluster-Based routing protocols

There is also a family of routing protocols that use network clustering in order to achieve better scalability in mobile ad hoc networks. CEDAR[70] and CBRP[35] are examples of the cluster-based routing protocols. Here nodes negotiate a topological partitioning of the network in a distributed manner, without a central coordinator. Generally, nodes are partitioned in clusters, and the membership of clusters changes as network topology changes. There are special nodes in the clusters that have a special role in the routing process. Their role can be for instance a "cluster-head" or a "gateway" between two clusters. Significant resources are needed to impose topological structure on a highly dynamic mobile ad hoc network. CEDAR uses the concept of a minimum dominating set to do cluster partitioning. This is the minimum set of nodes (dominating nodes) such that all nodes are at most one-hop away from a dominating set. When a source does not have a route to the destination it sends a route request packet to its dominating node. A dominating node is in charge of discovering a "core" path, or source route from the dominating node of a source to the dominating node of the destination. Once the core path is returned to the source, it is used for data traffic.

In CBRP, each node maintains the two-hop topology information to define clusters. Each cluster includes an elected cluster-head, with which each cluster member has a bidirectional link. Each cluster-head has a knowledge of the gateway nodes. Those are nodes that are at the border with another cluster. When a source has not a route to the destination, it sends a route request to its cluster-head. The route request is broadcasted in a controlled way, until it reaches the destination. When the request reaches the destination, it contains a loose source route specifying a sequence of clusters. Along its way back to the source, each cluster-head writes a complete source route into the route reply, based on its knowledge of cluster topology. The source receives the complete source route information, which it uses for data traffic.

3.3 Position-based routing protocols for Mobile Ad Hoc Networks

In this section we give an overview of routing protocols that use $geographical$ positions of nodes in the network for making packet forwarding decisions . We call this family of routing protocols *position-based* routing protocols. In 3.3.3 we identify the problem with the existing position-based routing protocol, and how this impacts our work.

The justification for applying position-based routing methods in mobile ad hoc networks was provided by the recent availability of small inexpensive low power GPS [80] receivers and techniques for finding relative coordinates based on signal strengths, and the need for the design of power efficient and scalable networks. A number of such algorithms were developed in the last few years, in addition to a few basic methods proposed about fifteen years ago.

Traditional MANET protocols (e.g., DSDV, DSR, AODV, ZRP) are designed to generate less routing protocol traffic in the face of a changing topology than traditional routing protocols in the fixed Internet. Nevertheless these protocols compute shortest-path routes either by using topological information of the whole network (e.g., DSDV), or topological information of the set of available routes between sources and destinations. In on-demand routing protocols, a node floods the network to discover a path when it is needed and caches it for later use. However, flooding is expensive, and the lifetime of a cached path is very short if the nodes move with a considerable speed. Traditional MANET routing protocols require a large routing overhead, and therefore they do not scale well for larger mobile ad hoc networks with high mobility rate.

Position-based routing protocols make use of the geographical information in order to achieve one or several of the following objectives:

- to avoid flooding of the network with the routing protocol traffic by reducing the propagation of control messages only to the selected geographical regions;
- to reduce intermediate system functions;
- to use geographical information for making packet (geographic forwarding) forwarding decisions.

Position-based routing protocols require that every node knows its own position obtained, for instance, by the GPS positioning system. In addition, propagation of some topological information is needed: as it will be presented below, there are protocols where each node need only know immediate neighbours' positions and the position of the destination to make forwarding decisions.

Localization of the topological information that must be communicated among nodes in the routing protocol improves the scaling of routing in three ways:

- it reduces the absolute volume of routing protocol message traffic;
- it reduces the size of the state that must be stored at nodes;
- it reduces the risk that a state stored at a node concerning a far-away portion of the network will become stale.

We can distinguish four main classes of existing position-based routing protocols:

1. Basic distance, progress and direction based methods make use of neighbours' positions for making packet forwarding decisions, but do not guarantee packet delivery (described in Section 3.3.1).
2. Routing protocols with guaranteed packet delivery make use of local neigbours' positions for making packet forwarding decisions, and these protocols guarantee packet delivery in static networks (described in Section 3.3.2)
3. Partial flooding routing protocols use nodes' positions to perform the controlled flooding in the network for packet delivery or controlled propagation of routing protocol messages (described in Section 3.3.4)
4. Hierarchical routing protocols (described in Section 3.3.5) make use of hierarchy of geographical regions to perform routing.

3.3.1 Basic Distance, Progress, and Direction Based Methods

In this class of routing protocols every node keeps information about the identity and position of its immediate neighbours. There are different strategies a node can use to decide to which neighbour a packet should be forwarded.

distance based schemes In this class of routing protocols the greedy method is used: the packet is forwarded to the neighbour that is closest to the destination. The source of the packet stamps the destination position in the packet.

The first proposed protocol in this class is the one proposed by Finn [21] in 1987. When none of neighboring nodes is closer to the destination than a current node C, Finn proposes

to search all n-hop neighbors (nodes at a distance at most n hops from the current node, where n is a network dependent parameter) by flooding the nodes until a node closer to destination than C is found.

A variant of greedy algorithms, called GEDIR, is proposed in [71]. In this variant, the message is dropped if the best choice for a current node is to return the message to the node the message came from.

progress based schemes The notion of progress is the key concept of several position-based methods proposed in 1984-86. Given a transmitting node S and receiver A, the progress is defined as the projection of the line connecting S and A onto the line connecting S and the final destination. Node A is in a forward direction if the progress is positive (for example, for transmitting node S and receiving nodes A, C and F in Figure 3.1); otherwise it is said to be in a backward direction (e.g. nodes B and E in Figure 3.1).

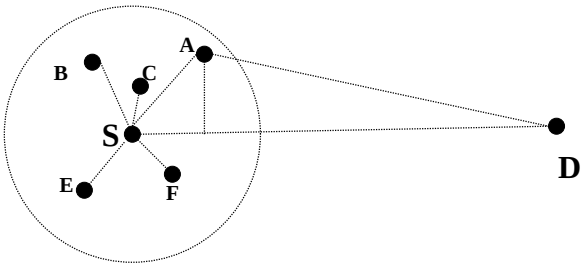


Figure 3.1: Illustration of positive and negative progress: nodes C, A, F are in forward direction, while nodes B and E are in backward direction. The circle around S indicates the maximum transmission range of S

Below are presented three progress-based schemes.

The first is the *Most Forward within Radius* (MFR) scheme [73]. In MFR, the packet is sent to the neighbor with which the greatest forward progress is attained (e.g. node A in Figure 3.1). MFR tries to minimize the number of hops to reach the destination. MFR is proved to be a loop-free algorithm [72]. MFR is a good strategy in scenarios where the sender cannot adapt the signal strength of the transmission to the distance between sender and receiver.

The second progress-based scheme is called Nearest with Forward Progress (NFP) [31]. NFR assumes that the sender can adapt its signal strength. With NFP, the packet is forwarded to the nearest neighbour of the sender that is closer to the destination. The benefit of NFP compared to MFR is that the probability of collisions is reduced significantly.

The third progress-based scheme is the $random$ progress method [55], packets destined to D are routed with equal probability towards one intermediate neighboring node that has positive progress. The rationale for the method is that, if all nodes are sending packets frequently, the probability of collision grows with the distance between nodes (assuming that the transmission power is adjusted to the minimal possible), and thus there is a trade-off between the progress and transmission success.

direction based schemes The compass routing method [44] is an example of direction based method. A source or intermediate node A uses the location information of the destination D to calculate its direction. Then the packet is forwarded to neighbor C, such that the direction AC closest to the direction AD. This process is repeated until the destination is, eventually, reached.

Direction-based routing is applied as part of other routing schemes. For instance, in [91] each node proactively maintains link-state routes to all nodes within a local zone. If node C finds the destination in its local zone, the packet is forwarded to the destination using the link-state intra-zone routing. If the destination is not in a local zone, C picks a node B on the boundary of its the local zone with the minimal angular distance from from the line joining C and the destination; intra-zone routing is used to forward the packet from C to B. B repeats the same procedure until the packet arrives at the destination. [91] also proposes a scalable location management: as the packet progresses toward the destination, the closer nodes increase the accuracy of the destination position.

[72] compares the performance of the aforementioned methods by simulations. It is shown that MFR and GEDIR protocols, in most cases, provide the same path to the destination. Simulation in [72] revealed that nodes in GEDIR and MFR methods select the same forwarding neighbor in over 99% of cases, and, in the majority of the cases, the entire paths were identical. The hop count for MFR is somewhat higher than for GEDIR, while the success rate is similar. In densely populated networks, the shortest path between two nodes corresponds closely to the Euclidean straight line between them, and thus in such networks hop counts of greedy and MFR methods nearly match the performance of the shortest path algorithm.

[72] demonstrated that GEDIR and MFR are loop-free, while the direction based algorithms are not loop-free.

The advantage of the basic distance, progress and direction based methods is that the state required in nodes is independent of the total number of nodes in the network, but depends only on the density of nodes in the radio range of the emitting node. However, these methods do not

guarantee delivery of packets to the destination.

3.3.2 Position-based routing with guaranteed packet delivery

In this section we review position-based routing that guarantee packet delivery in static networks.

Geographical Routing Algorithm (GRA)

Geographical Routing Algorithm (GRA)[66] uses neighbours positions for greedy packet forwarding, however it uses route discovery to discover a route to destinations that can not be reached in a greedy way. In GRA every node has only a partial knowledge of a network. It knows about its immediate neighbors and a small number of remote nodes to which it has discovered a path. When an intermediate node receives a packet to forward, it checks which of the nodes that it knows is closest to the destination. Then the packet is forwarded to the neighbour that is the next hop towards the node that is closest to the destination. Each node thus forwards the packet in the similar way till the packet reaches the destination. If it happens that some node S does not know about any node that is closer to the destination D than itself, a route discovery method is invoked. This can be either breath first discovery using flooding or depth first search. A route discovery method should find an acyclic path from S to D. All intermediate nodes on the path keep in the routing table the next hop in order to reach D. During a route discovery, all intermediate nodes keep the next hop information in order to reach the given destination. The drawback is that, whenever a single link in a route is broken, the route should be rebuilt.

Greedy and Perimeter Mode Packet Forwarding

The GPSR[39] and GFG[11] protocols use a greedy method for making packet forwarding decisions, however unlike GEDIR, they ensure packet delivery in the case of static networks. Packet forwarding decisions are made using only information about a node's immediate neighbours and the location of destination. It is forwarded in the greedy way to the neighbour that is closest to the destination.

As it was already mentioned, the greedy packet forwarding suffer from the so-called *local minimum phenomenon*. This happens when a packet is stuck at the node that does not have a closer neighbour to the destination. This situation indicates that there is a hole in the geographic distribution of nodes. Illustration of greedy routing failure is given in Figure 3.2.

To deal with this problem, when the greedy method is unable to deliver the packet, GPSR and GFG use a planar subgraph of the wireless network's graph to route around the perimeter of

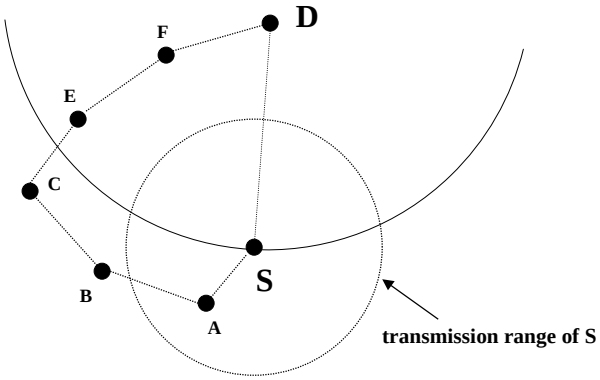


Figure 3.2: Illustration of greedy routing failure: S does not have a neighbour that is closer to D. However, there exists a path from S to D. The path is given by the list of nodes $\{$A,B,C,E,F$\}$.

a hole. This method is first proposed by Bose *et al.* in [11]. Packet forwarding for such a packet is switched from greedy to the $perimeter$ mode. The knowledge of identities and locations of its one-hop neighbours is sufficient for a node to determine the edges of the planar subgraph. Packets that are in perimeter mode are forwarded using a planar graph traversal. As soon as a packet reaches the node that is closer to destination than the node that initiated perimeter-mode forwarding, a packet is then forwarded in a greedy way. For example, in Figure 3.2, at A packet is forwarded in perimeter mode through nodes B and C. At E, since E is closer to D than A, packet is again forwarded in greedy mode.

In the following, we present elements of perimeter-mode packet forwarding: namely, graph planarization algorithm, and planar graph traversal method.

wireless network planar subgraph A widely accepted basic graph-theoretical model for a wireless network is the $unit$ graph model, defined in the following way. Two nodes A and B in the network are neighbors (and thus joined by an edge) if the Euclidean distance between their coordinates in the network is at most R, where R is the transmission range that is equal for all nodes in the network.

Starting from unit graphs, there are several algorithms for construction of a planar subgraph of the unit graph. A graph in which no two edges cross is known as $planar$. Here, we present one of them, called the Gabriel Graph (GG) [23].

We assume that nodes in the network have a negligible difference in altitude, so that they can be considered roughly in the plane. The Gabriel graph is a spanning subgraph of the original unit graph. It is defined as follows: given any two adjacent nodes u and v in the

network (that is, $(u, v) \leq R$), the edge (u,v) belongs to the Gabriel graph if and only if no other node w of the network is located in the disk with the segment (u,v) as its diameter. Figure 3.3 depicts the rule for construction of the GG. The shaded circle between u and v, must be empty of any witness node w for edge (u,v) to be included in the GG. When we begin with a connected unit graph and remove edges that are not part of the GG, we cannot disconnect the graph. (u,v) is only eliminated from the graph when there is a w within range of both u and v, and thus this is node via which u and v are connected. Figure 3.4 illustrates one original graph and its planar Gabriel subgraph.

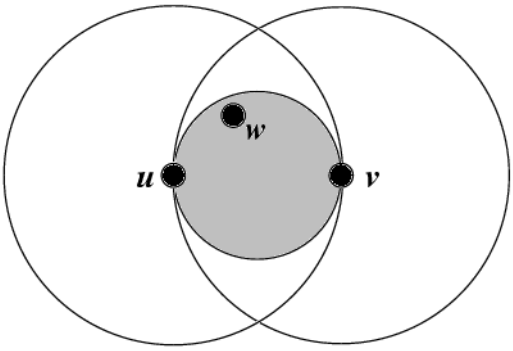
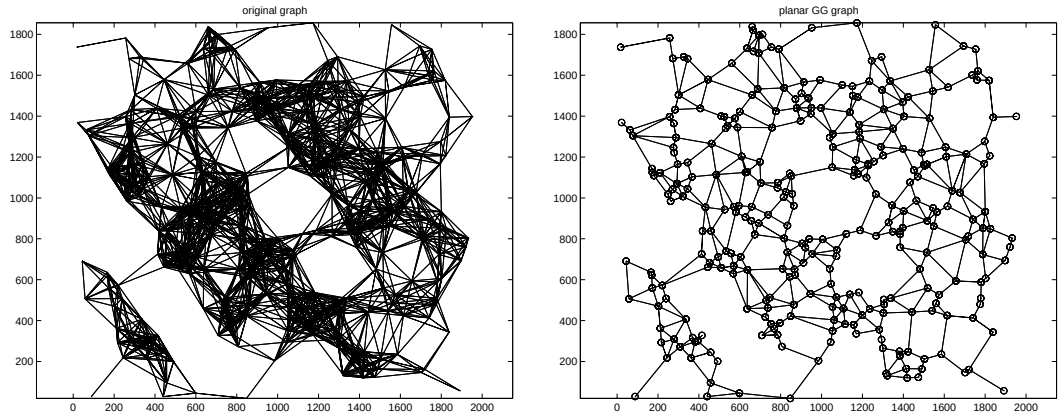


Figure 3.3: Figure presents how the Gabriel graph is built. For edge (u,v) to be included, the shaded circle must include no witness w. Gabriel Graph in this case should not include edge (u,v).

The Gabriel graph is planar, that is, no two edges of it intersect each other [11]. Once the Gabriel graph is extracted from the network, routing is performed along its edges. Its planarity and its connectivity ensure message delivery by routing along the faces of the graph.

The Relative Neighbourhood Graph (RNG)[78] is another well-known planar graph. RNG builds the planar subraph in a way similar to the Gabriel graph. The RNG is a subset of the GG.

The recent proposed planar spanner of a wireless network graph is the Restricted Delaunay Graph (RDG) [24]. Combined with a node clustering algorithm, RDG can be used as an underlying graph for geographic routing. RDG guarantees that between any two nodes there exists a path in the RDG whose length, whether measured in terms of topological or Euclidean distance, is only a constant (so-called stretch factor) times the optimum possible length. RDG has better spanning property than GG or RNG.



(a) (b)

Figure 3.4: (a) Original connectivity graph (300 nodes, placed uniformly at random on a 2000-by-2000-meter region; radio range 250 meters), (b) Gabriel graph - planar subgraph of the original connectivity graph

Bose et al.[10] proved that the Euclidean stretch factor of GG and RNG are $\Theta(\sqrt{n})$ and $\Theta(n)$, respectively, where n is the number of nodes. GG and RNG favor short edges in the graph. As the consequence, when the density of the node set is high, we may have to traverse many short edges in the graph in order to connect two nodes.

The solution for construction of a planar subgraph of a non-unit graph (non-uniform radio ranges) is presented in [6].

Once the planar subgraph of the wireless network is obtained, it is used as the routing graph where the planar graph traversal method is performed.

planar graph traversal method The planar graph traversal method was originally proposed by Kranakis *et all* [44]. Here we present the modified version of the original algorithm, as used in GFG [11] and GPSR [39].

A connected planar subgraph partitions the plane into $faces$ that are bounded by polygonals made up of edges of the planar subgraph. When a packet whose destination is D enters perimeter mode at node X, the packet is forwarded on progressively closer faces of the planar graph, each of which is crossed by the line $\overline{XD}$. On each face, the traversal uses the righthand rule to reach an edge that crosses line $\overline{XD}$. Once such an edge is encountered, the traversal moves to the adjacent face crossed by $\overline{XD}$. An example of planar

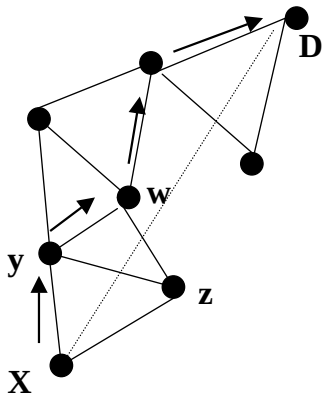


Figure 3.5: The planar graph traversal method: D is the destination; X is the node where the packet enters perimeter mode. Solid arrows are forwarding hops.

graph traversal is presented in Figure 3.5. Here, X first sends the packet to y following the right-hand rule (the packet is forwarded to the first edge counterclockwise about X from line $\overline{XD}$). y receives the packet, and y borders the edge (y,z) that intersects the $\overline{XD}$. There, the packet is forwarded along the next face bordering the edge (y,z). y forwards the packet along the first edge of this next face (y,w) by the right-hand rule. In the same way, the packet is forwarded until the face containing D is reached. If, however, the destination cannot be reached, (e.g., when the network is disconnected), during its journey along faces of the planar graph, the packet will return to the first edge where its touring began. In static networks, this is the sign that the packet cannot be delivered to the destination, and thus may be dropped.

In a static network, planar graph traversal method guarantees delivery of packets from any source to any destination [11]. However, paths where only this method is used can be extremely long [11]. Therefore, [11] and [39] propose the packet to be forwarded in the greedy mode, and only when this fails, to switch the packet to perimeter mode and use the planar graph traversal method.

We present a simple evaluation of the effects of node density on the success rate of the greedy packet forwarding. We generate random unit graphs with different average node density. The simulation area is 1km^2, and the transmission range is 250m. In each simulation, nodes are distributed in the simulation area according to the two-dimension Poisson distribution. A node degree is given as the number of nodes that are placed within its transmission range. Different node densities are used to ensure different average node degree. Generated graphs that are disconnected are ignored. Figure 3.6 reports, for each value of the node degree, the fraction of

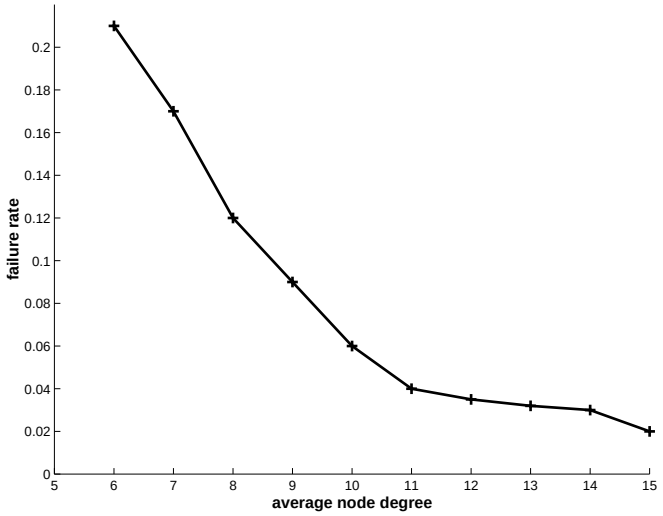


Figure 3.6: Fraction of paths that cannot be found in a greedy way as a function of the average node degree

paths that cannot not be found by the greedy forwarding. Paths between every two nodes are counted. We can conclude from Figure 3.6 that greedy packet forwarding works well when the node degree is larger than 10.

Thus in a dense network, packets are normally forwarded in a greedy way, and the perimeter mode is used occasionally when a packet is stuck because a node does not have a one-hop neighbour that is closer to the destination. Then a packet is forwarded in perimeter mode for only a very few (2-3) hops, before a node closer than the point of entry into perimeter mode is reached, and then greedy forwarding resumes. This is illustrated in Figure 3.2, where the packet is forwarded in the perimeter mode from S to E. Since E is closer to D than S, at E the packet is again forwarded in the greedy mode.

On sparser networks, the perimeter mode tends to be used for longer sequences of hops [38].

3.3.3 A problem with perimeter-mode packet forwarding in networks with dynamic topologies

Note that, several routing schemes for mobile ad hoc networks, have proven that they guarantee message delivery (and that it is loop free) in the static case. However, few routing protocols are loop-free in network with dynamic topology (e.g, DSR ensures loop-free packet delivery, since it uses loop-free source routes).

GPSR and GFG both guarantee packet delivery in static networks. However, in mobile networks, when the packet is in the perimeter mode, loops can happen. In GPSR papers [39, 38], authors experimented with mobile nodes and compared the performance of GPSR against DSR in the networks of variable size, using the NS [1] simulator. They demonstrated a better performance of GPSR, over DSR, in larger dense networks (nodes have on average more than 15 neighbours). In dense networks most packets are forwarded in the greedy mode, where greedy forwarding well approximates shortest paths. [38] just mentions the looping problem in sparser networks. However, the problem is not analyzed, nor evaluated.

We have implemented perimeter-mode of packet forwarding in GloMoSim [74], as a part of terminode routing (described in Chapters 4 and 5). Here, we report on the problems of loops that were encountered while performing the simulations.

Figure 3.7 presents one example of a situation where loops are possible. In this example the packet is already in the perimeter mode when it arrives at node 1; the packet is forwarded in the current face in the clockwise order. Step 1: neighbours of node 2 are nodes 1, 3 and 4. Node 5 was not included in the list due to whatever cause (it may be for example far from 2). Then, node 2, because of the right-hand rule, will send the packet to node 3. Step 2: Node 3, which has nodes 2 and 4 in its neighbours list, sends the packet to node 4, again because of the right-hand-rule. Step 3: Node 4, which has nodes 3 and 6, and now node 5 in its neighbours list (node 5 may have just arrived in 4's radio range) calculates planar graph and sends the packet to node 5. Step 4: Node 5, which now has nodes 4 and 2 in its neighbours list, will send the packet to node 2. Node 2 now has nodes 5, 1 and 3 in its neighbours list. The right-hand-rule will give node 3 as the next hop. If node 5 stops moving, or stays in node 2's radio range for a while, the packet will always follow the same path: 2 - 3 - 4 - 5 - 2 - ...

In this example, the reason for the loop is as follows. The packet circulates along edges of the planar face (edges (1,2), (2,3), (3,4)). Then, the topology is changed and node 5 arrives closer to nodes 4 and 5 and thus a new face is created. This face consists of nodes (2, 3, 4, 5) and does not cross the line between the node where the packet has entered into the perimeter mode and the destination. The packet is blocked inside the newly created face. The loop will disappear if the topology changes again and the face reopens or it moves to cross the line to the destination (and thus enters in the new face). If this does not happen the packet may loop until its time-to-live maximum hop counter reaches zero, where it is dropped.

We evaluate by using simulations the loop problem perimeter-mode of packet forwarding.

Simulations are performed in GloMoSim with the following simulation parameters[1]: uniform

[1] all used simulation parameters are described in details in Section 5.3

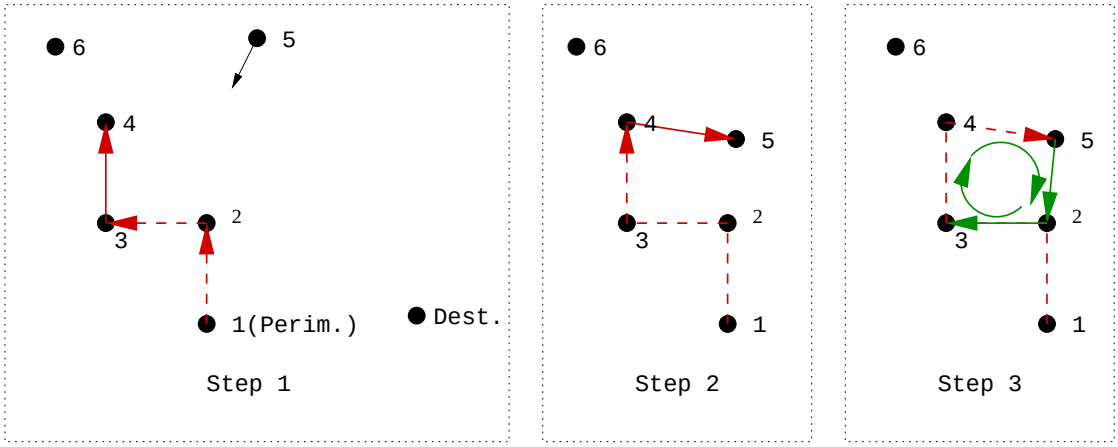


Figure 3.7: Figure illustrates one example of loops in the perimeter mode of packet forwarding

node placement the rectangular area of size 2500m X 1500m; the random waypoint mobility model with the speed uniformly distributed in interval 0-20 m/s and pause time of 100 seconds. A beacon interval ("hello" packets) of 1 second, a timeout of 2 seconds after which a neighbour in the neighbours list is deleted if it is not updated. All nodes have the same transmission range equal to 250 meters. 40 CBR (constant bit ratio) applications, with a packet size of 256 bytes emitted each second (ie. a throughput of 2kbits per source flow).

The results focus on the effect of the node density in the network. This latter parameter is very important, as it determines if the perimeter mode will be often used or not.

Figure 3.8 shows how many applications require that at least one packet originated by them has to use the perimeter mode forwarding to reach the destination. When the node density is low, many packets are not arriving at their destination due to disconnected network. At a specific node density (1 node in 37500 m^2, or 100 nodes in our simulations), all the applications required that at least one packet originated by them had to use perimeter packet forwarding to reach their destination. This node density seems to be the threshold between an often disconnected graph and a graph where greedy forwarding is possible. For higher densities, the perimeter mode is not used much (because most of the time greedy forwarding is possible). If it is used, a packet is forwarded in perimeter mode for only a very few (2-3) hops, before a node closer than the point of entry into perimeter mode is reached, and then greedy forwarding resumes.

Figure 3.9 shows the average number of packets dropped, due to a disconnected network or due to loops. We see that about one third of all the packets dropped is due to loops in the perimeter mode, which is quite an unexpected figure. While dropping a packet because of a disconnected

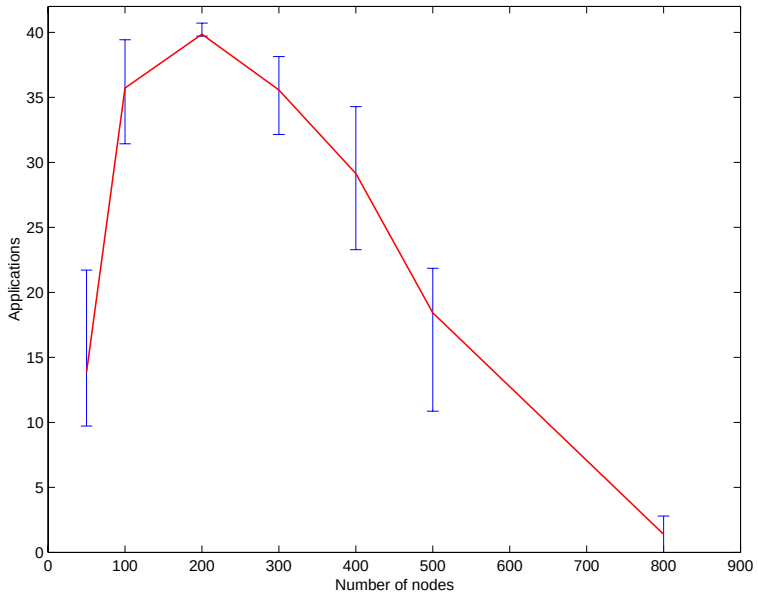


Figure 3.8: The number of applications which use the perimeter-mode packet forwarding. The error intervals represent the min. and max. values

network is harmless for the network, with regard to congestion, having many packets dropped because of loops may overwhelm the network, as they each use *ttl* hops.

We may conclude that perimeter-mode packet forwarding can be used as a repair mechanism whenever the greedy forwarding fails. However, due to loop problems that may occur in mobile networks, perimeter packet forwarding should be avoided as much as possible. In Chapter 4, we present the terminode routing that conducts the packet to traverse regions with good node density, and uses mostly greedy packet forwarding. Terminode routing tries to avoid regions with low node density, where perimeter packet forwarding is to be used.

3.3.4 Partial Flooding Algorithms

Location Aided Routing (LAR)[92] does not propose to use position information for making packet forwarding decisions, but to use positions to enhance the route discovery phase in DSR [12]. LAR uses location information to reduce the search space for a desired route. Limiting the search space results in fewer route discovery messages. When node S wants to find the route to node D, S computes an *expected zone* for D based on last known location and velocity of D. S then determines a *request zone*, as a set of nodes that should forward the route request packet. LAR proposes two Location-Aided Routing schemes for route discovery. In LAR scheme 1, the

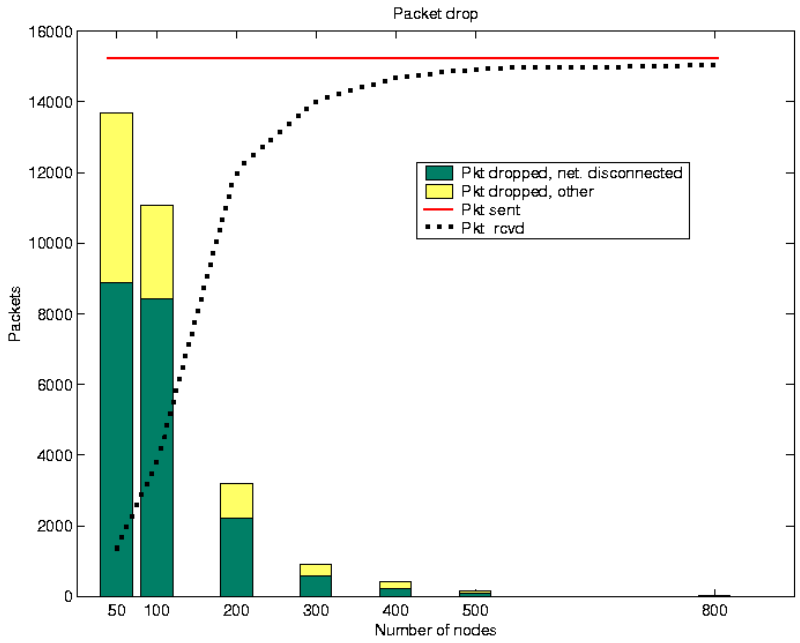


Figure 3.9: Figure presents the average number of received packets and the average number of packets dropped due to the loop problem in the perimeter-mode packet forwarding. Different number of nodes are observed.

request zone is a rectangular geographic region (see Figure 3.10). In LAR scheme 2, the source or an intermediate node forwards the packet to all nodes that are closer to the destination than themselves. With LAR, end-to-end routes are still DSR's source routes.

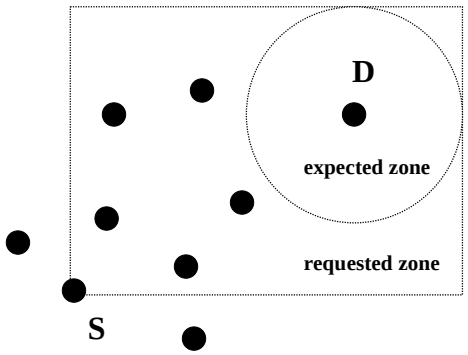


Figure 3.10: Example of the expected and requested zones in LAR scheme 1

Location Distance Routing Effect Algorithm for Mobility (DREAM) [7] is a routing protocol in which the information about the location and the speed of the destination is used to obtain

the expected direction of the destination. Node S that has a packet to send to destination D determines the direction of the destination: direction is defined by the tangents from S to the circle centered at D and with radius (r) equal to a maximal possible movement of the destination since the last known D's location (see Figure 3.11). S forwards the packet to all neighbors that lie in the direction of D. DREAM also proposes how to disseminate location information in the

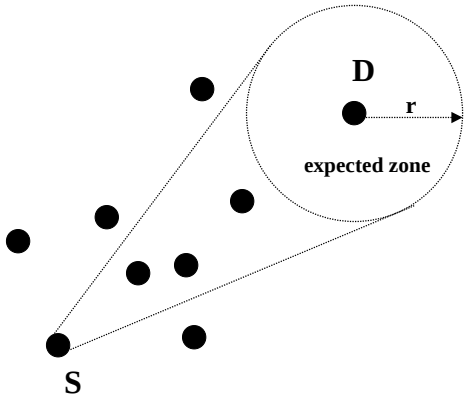


Figure 3.11: Example of the expected zone in DREAM

network in the scalable way. In DREAM, each node periodically exchanges control packets to inform all other nodes of its location. Each control packet is assigned a life time based on the geographical distance from the sender. DREAM sends short lived packet more frequently than long lived packets 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.

3.3.5 Hierarchical Geographic Routing

In the zone-based hierarchical link state (ZHLR) [36] protocol the network is divided into non-overlapping zones. Each node knows the node connectivity within its zone (using a limited link-state routing protocol within the zone), and the inter-zone connectivity of the whole network. The source send the packet to the destination in its zone using the proactive intra-zone link-state protocol. Otherwise, the sender initiates the search for the destination by sending route request, one to each neighbouring zone. The zone that contains the destination (more precisely, the first node from that zone reached on the way to the center of that zone) replies with the exact coordinates of the destination back to the sender node. Once the sender learns the zone in which is the destination, it puts this information in the packet. The packet is forwarded to the destination zone using the inter-zone path, and once the packet reaches the zone of the destination, the packet

is forwarded using the intra-zone path.

3.4 Conclusion

In this chapter we gave the overview of the existing routing protocols for mobile ad hoc networks. Traditional MANET routing protocols were reviewed, as well as position-based routing protocols for mobile ad-hoc networks. The latter use geographic positions of nodes and of packet's destination to make forwarding decisions. Greedy and perimeter mode packet forwarding were described. We presented and evaluated the problem of loops that can appear when the perimeter-mode packet forwarding in used.

Chapter 4

Terminode Routing

4.1 Introduction

In this chapter we focus on the problem of routing in a large mobile ad hoc network that we call a "terminode" network. A terminode network is a wide area, large, totally wireless, mobile network. We call nodes in this network, $terminodes$, because they act as network nodes and terminals at the same time.

The Terminodes Project [2] is a long-term research project at the Swiss Federal Institute of Technology, Lausanne. The Terminodes Project investigates terminode networks and covers all layers (from the physical to the application). Here we cover the routing aspects of this project. Terminode network is a potentially very large mobile ad hoc newtork. Therefore, the target of our work is different from MANET[50] proposals that focus on smaller ad hoc networks consisting of up to several hundreds of nodes.

The routing solution in a terminode network is designed with three requirements in mind:

- it should scale well in a relatively large mobile ad hoc network;
- it should cope with dynamically changing network connectivity owing to mobility;
- terminodes need to be highly cooperative and redundant, but, most of all, they cannot use complex algorithms or protocols that would require a high routing overhead.

This chapter describes elements of single path terminode routing. In a highly mobile ad hoc network paths can easily be broken or become congested. As a response to such uncertainty in the network, we advocate that routing in a large self-organized ad hoc network should be multipath. At the end of this chapter we also describe the architecture of how to apply multipath terminode routing.

4.2 Addressing Issues and Assumptions

Each terminode has a permanent End-system Unique Identifier (EUI), and a temporary, location-dependent address (LDA).

EUI is a 64 bit long address burnt-in the hardware and mappable to an IPv6 address.

LDA is simply a triplet of geographic coordinates (longitude, latitude, altitude). We assume in this work that all nodes in the system are equipped with some hardware that provides them with their current location. The location information can be obtained either by means of the Global Positioning System (GPS), or, if GPS is not available (e.g., indoors), the GPS-free positioning methods ([14], [65], [29]) can be used. Coding longitude and latitude with an accuracy of 10^{-4} grades gives a position accuracy of about 10 meters. With this accuracy, the total number of position triples is to the order of 10^{16}, thus an LDA could be coded with 48 bits.

We intend to reserve a portion of the IPv6 addressing space for terminodes; the IPv6 address of a terminode is then algorithmically mapped from its EUI. From an IPv6 viewpoint, the set of terminodes is one huge subnetwork. Two terminodes normally use the TCP/IP protocol stack to communicate; however, inside the network of terminodes, packet forwarding does not use IP addresses, similar to how bridges operate in a large bridged network.

We assume that there is a location management that enables terminodes in the network to determine approximate locations of other terminodes. Location management in a terminode network is performed by a combination of the following functions. Firstly, a location tracking algorithm is assumed to exist between terminodes when they have successfully established communication; this allows communicating terminodes to continuously update the the corresponding LDA information. Secondly, a location discovery service is used to obtain a probable location of terminode B (LDA_B) that A is not tracking by the previous method.

We assume that terminodes move with either a speed of a pedestrian or a car, such that we can obtain (with the location management) the destination location with the precision of approximately one transmission range and with the validity of about ten seconds. At the end of this chapter, in Section 4.11, we give the justification for the requirements of terminode routing on location management.

Even if our protocol uses geographic locations, it is independent from the physical infrastructure (i.e, it does not assume directional antennae) and from the physical underlying layer. We further assume bidirectional radio reachability. We assume the existence of a MAC protocol that supports link-level acknowledgements for unicast packets. An example of such a protocol is the IEEE 802.11 MAC protocol, which we used in our simulations to evaluate routing in a terminode network. We consider topologies where terminodes move in a two-dimensional plane. We also

assume that a terminode network is most of the time connected, although temporary partitions can occur.

4.3 Terminode Routing - has two components: TLR and TRR

Recall from the previous chapter that position-based routing scales better for large mobile ad hoc networks than routing protocols that do not use nodes' positions for making packet forwarding decisions. In position-based routing protocols sources should know destinations positions accurately enough at all times in order for packets to reach their destinations. However the location management service may not provide accurate location information at all times. This is especially true if the nodes are close and their relative positions change frequently. In this case positional errors and inconsistent location information may result in the packet circulation around the destination's position that is known at the source, while the destination may have moved away from this position.

Because terminode routing should scale well in a relatively large mobile ad hoc network, it is a position-based routing protocol. However, it has a mechanism to cope with the problems due to position inaccuracy. Terminode routing is a combination of two routing protocols: Terminode Local Routing (TLR) and Terminode Remote Routing (TRR).

TRR is used to send data to remote destinations and uses geographic information; it is the key element for achieving scalability and reduced dependence on intermediate systems. When the packet gets close to destination, the packet forwarding method switches to TLR. TLR is a mechanism that allows for destinations to be reached in the vicinity of a terminode and does not use location information for making packet forwarding decisions. It uses local routing tables that every terminode proactively maintains for its close terminodes. TLR is a strategy for handling the destination position deviation due to mobility. Once TLR is started, packet forwarding cannot revert to TRR.

4.4 Terminode Local Routing (TLR)

Terminode Local Routing (TLR) is used by terminodes to proactively learn about terminodes in their vicinity, and for packet forwarding to these terminodes.

TLR includes characteristics of MANET routing protocols. Several MANET protocols were considered for TLR. As said in Chapter 3, AODV [59] and DSR[37] build routes on demand, however these protocols do not proactively maintain information about nodes in the network.

Therefore, these protocols are not considered for TLR. With proactive protocols, nodes proactively maintain identities and routes to other nodes in the network, and this characteristic is more suitable for TLR. However, as said in Section 3.2.1, proactive protocols do not scale well for larger mobile ad hoc networks. The reasons is a high routing load. Our goal is to use the proactive procedure for TLR within a limited scope. A similar approach is used by the intrazone routing protocol (IARP) in ZRP[62].

The TLR-reachable area of S includes the terminodes whose minimum distances in hops from S are at most equal to the *local radius*. The local radius is the measure, in number of hops, of the TLR-reachable area. With TLR, every terminode maintains routing information to those terminodes (that are called *TLR-reachable*) that are no more than a local radius hop away. TLR is a link-state routing protocol limited within a scope of a TLR-reachable area. In the current implementation of TLR, all terminodes have the same *local radius* equal to two hops.

Now we describe the two methods of TLR: (1) the building of local routing tables, and (2) TLR packet forwarding.

1. Building of TLR routing tables

   Each node keeps in its routing table the EUI and LDA information of its immediate neighbours, as well as the EUI information about its two-hop distant terminodes. The EUI information of immediate and two-hop distant terminodes is used for TLR packet forwarding. The LDA information of immediate neighbours is used in TRR for sending packets to nodes out of the TLR-reachable area, as explained in the next section. Each node periodically advertises by means of HELLO messages its current set of immediate neighbours. HELLO messages are periodically broadcasted at the MAC layer. A terminode announces in a HELLO message its own EUI and LDA, as well as EUIs of its immediate neighbours. Upon reception of a HELLO message, a node updates its local routing table. A node maintains its routing table in which it keeps the information about its one-hop neighbours, and its two-hop distant terminodes that these one-hop neighbours give access to. The information is recorded in the routing table as an entry. An entry consists of the following fields: the EUI and LDA of the one-hop neighbour and EUIs of two-hop distant terminodes. Each entry has an associated holding time. If a node does not hear from its neighbour for some amount of time, it removes from the routing table the entry that corresponds to the lost neighbour, as well as all two-hop distant terminodes that were reachable via the lost neighbour.

   TLR does not specify that HELLO messages carry LDAs of the sender's immediate neighbours. However, this information can be easily provided, such that nodes learn about LDAs

of two-hop distant terminodes. This information may be useful for TRR, as it is described in Section 4.9.1.

We assume the existence of bidirectional links in the network, then, when node A receives a HELLO message from node B, A can reach B.

2. TLR packet forwarding

   When the source, or an intermediate node finds that the destination is TLR-reachable, the "use TLR" bit in the packet header (see Figure 4.4) is set to one, if not already set. This is the sign that from now on, the only mechanism used to forward the packet is TLR. If the destination is two-hops away, the next-hop to send the packet to is determined from the routing table. This is the one-hop neighbour via which a two-hop distant terminode can be reached. If a two-hop distant terminode can be reached via several one-hop neighbours, we choose the one-hop neighbour whose entry is updated most recently. Otherwise, if the intermediate node receives the packet whose "use TLR" bit is already set to one, the packet should be sent directly to the destination; if the intermediate node does not find the destination among its one-hop neighbours, the packet is dropped. This ensures that TLR is loop-free.

   Figure 4.1 presents the TLR packet forwarding in pseudocode.

X has packet p to forward to D with TLR:

if ($p.use\_tlr\_bit = 0$) $p.use\_tlr\_bit := 1$
if ($EUI_D \in X.TLR\_routing\_table$)
    transmit($p, X.TLR\_routing\_table.next\_hop(EUI_D)$)
else drop p

Figure 4.1: The Terminode Local Routing (TLR) packet forwarding algorithm (pseudocode)

We also may use TLR within an area with a local radius larger than two hops. The larger the local radius is, the more robust the location-based routing is to location inaccuracy. However, maintaining a larger TLR-reachable area results in more routing overhead and routing converges slower at a higher mobility. On the one hand, a small TLR-area guarantees the routing information is maximally updated even if nodes move with a high speed, at the minimum routing overhead cost. On the other hand, TLR should be large enough to cope with the location inaccuracy problem.

In Section 5.4 we verified by using simulations, that when the local radius is two, routing is robust against location inaccuracy to a satisfactory extent. At the same time keeping the TLR-reachable area is done at a minimal cost. In terminode routing, the knowledge of immediate neighbours is necessary for TRR. For this aim, HELLO messages are periodically broadcasted by all nodes in the network. The only additional requirement for TLR is that a sender of a HELLO messages also includes in a HELLO message also its immediate neighbours.

4.5 Overview of Terminode Remote Routing (TRR)

This section brings the overview of all elements of Terminode Remote Routing (TRR). In the following sections, these elements are described in details.

TRR allows data to be sent to *non-TLR-reachable* destinations. Its default method is *Geodesic Packet Forwarding (GPF)*. GPF is basically a greedy method that forwards the packet closer to the destination location until the destination is reached. GPF is described in Section 4.6.

We propose a method (described in Section 4.10) that enables the source to estimate whether GPF is successful in forwarding data to the destination. If this is not the case, TRR primarily forwards packets on *anchored paths*. In contrast with traditional routing algorithms, an anchored path does not consist of a list of nodes to be visited to reach the destination. An anchored path is a list of fixed geographic points, called $anchors$. In traditional paths made of lists of nodes, if nodes move far from where they were at the time when the path was computed, the path cannot be used to reach the destination. Given that geographic points do not move, the advantage of anchored paths is that an anchored path is always "valid".

In order to forward packets along an anchored path, TRR uses the method called *Anchored Geodesic Packet Forwarding (AGPF)* (described in Section 4.7). AGPF is a loose source routing method designed to be robust for mobile networks. With AGPF the packet is sent in the direction of an anchor, thus trying to reach some terminode in the proximity of this anchor. Thereon, the packet is forwarded in the direction of the next anchor on the anchored path. Anchored paths are obtained at the source by the path discovery methods. We propose two such methods: the first one is called Friend Assisted Path Discovery (FAPD), and the second is called Geographic Maps-based Path Discovery (GMPD). FAPD enables the source to learn the anchored path(s) to the destination using, so-called, $friends$, terminodes to which the source have already discovered paths. FAPD is described in Section 4.9.1. GMPD is another method for anchored path discovery, which assume that the network topology is known to all nodes in the network. GMPD is described in Section 4.9.2.

Source S sends packet p to destination D:

if ($EUI_D \in S.TLR\_routing\_table$)
  S applies TLR; //(see Figure 4.1)
else
  S acquires LDA_D
  if (S has anchored path(s) to D)
    S applies $AGPF$ //(see Figure 4.6)
  else if ($GMPD$ maps available $\vee$ $S.list\_of\_friends \neq \emptyset$)
    S starts path discovery ($FAPD$ or $GMPD$) // (see Sections 4.9.1and 4.9.2)
  else
    transmit($p, GPF(LDA_D).next\_hop$) //S sends p with GPF
    in the the direction of LDA_D (see Figure 4.5)

Figure 4.2: Packet Forwarding algorithm at the source (pseudocode)

X has packet p to forward to destination D:

if ($p.EUI_D = X.EUI$) then receive packet
else
  if ($p.use\_tlr\_bit = 1 \vee D \in S.TLR\_routing\_table$)
    X applies TLR //(see Figure 4.1)
  else if ($dist(X, LDA_D) < X.transmission\_range$)
    X expedites termination of TRR // (described in Section 4.8)
  else if ($p.anchored\_path \neq \emptyset$)
    X performs $AGPF$ //(see Figure 4.6)
  else transmit($p, GPF(LDA_D).next\_hop$); //S sends p with GPF
    in the direction of LDA_D (see Figure 4.5)

Figure 4.3: Packet Forwarding algorithm at an intermediate node (pseudocode)

Figures 4.2 and 4.3 present the global operation of terminode routing in pseudocode at the source and at an intermediate terminode. Figure 4.4 presents the packet header fields used in terminode routing.

4.6 Geodesic Packet Forwarding (GPF)

GPF is a simple method for sending data in the direction of a geographic point. This point can be an anchor (see AGPF below) or the destination location. Unlike TLR, GPF is based solely on

PACKET HEADER
Source EUI (EUI_S)
Destination EUI (EUI_D)
use_TLR_bit
Destination LDA (LDA_D)
anchored_path
additional fields used for GPF, TRR termination and FAPD

Figure 4.4: Terminode routing packet header fields (in unicast packets)

locations. A similar method is used in GFG[11] and in GPSR [39].

Source S uses GPF to send data to remote destination D in the following way. At first, S acquires some approximate value of the D's location (LDA_D), using a LDA management scheme. S stamps LDA_D within a packet header, thus, LDA_D serves as the reference geographic point towards which the packet is sent.

Then S sends packets with GPF in the $greedy$ manner: the packet is sent to some neighbour X within a transmission range of S where the distance to D is the most reduced. In turn, X checks whether D is TLR-reachable: if not, X sends the packet to its neighbour that is closest to the destination. Otherwise, X uses TLR to forward the packet.

In this simplest form, GPF will often not work. If there is no connectivity along the shortest line from S to D, due to obstacles or a terminode desert, then the method fails. The packet may be "stuck" at some terminode that does not have a neighbour that is closer to the destination. One possible solution to this problem is to use the method of a *planar graph traversal*, where a packet is routed around the perimeter of the region where there are no terminodes closer to the destination (this solution is also used in GFG[11] and GPSR [39]). Then, we say that the packet starts being forwarded in $perimeter$ mode. The packet is forwarded in perimeter mode until it arrives at the terminode that reduces the distance to the destination, and thereon the packet is forwarded in a greedy manner, as described above. We described the operation of perimeter-mode packet forwarding in Section 3.3.2, while its pseudocode is given in Appendix B.

The greedy-mode packet forwarding is a preferred mode of GPF. As already presented in Section 3.3.3, perimeter-mode forwarding is less robust against looping in mobile networks, and should be used only as a recovery when the greedy-mode forwarding is not possible. Greedy-mode forwarding is possible when there is a good connectivity along the shortest line from the source to the destination. However, this may not always work. In order to circumvent holes in terminodes distribution in a large area mobile ad hoc network, we introduce the method called

X has packet p to forward in the direction of location LOC with GPF:

if ($p.mode$ = $greedy$)
  //GPF.greedy_forward:
  choose $Y \in X.neighbours$ to minimize $dist(Y, LOC)$
  if ($dist(Y, LOC) < dist(X, LOC)$)
    transmit(p, Y)
  else
    //GPF perimeter forward (see Appendix B)
    $p.mode := perimeter$, $Y := GPF\_perimeter(LOC).next\_hop$, transmit($p$, Y)
else $Y := GPF\_perimeter(LOC).next\_hop$, transmit($p$,$Y$)

Figure 4.5: Geodesic Packet Forwarding (GPF) (pseudocode)

AGPF.

4.7 Anchored Geodesic Packet Forwarding (AGPF)

The key element of *AGPF* is the *anchored path*. The anchored path is a list of fixed geographical points, called $anchors$. Anchors are computed by source nodes, using the path discovery methods that are presented in Section 4.9. A source terminode adds to the packet the anchored path that is used as loose source routing information. With AGPF the packet is forwarded so that it loosely follows an anchored path. The sequence of intermediate terminodes on the way to the destination depends on the actual physical terminodes distribution in the plane.

AGPF works as follows. At the source, the packet is sent in the direction of the first anchor (AP1) on the anchored path by applying GPF: the source sends data to an immediate neighbour that has a smaller distance to AP1. When an intermediate terminode receives a packet with the anchored path, it checks whether AP1 geographically falls within its transmission range. If so, it deletes AP1 from the anchored path and sends the packet in the direction of the next anchor (AP2). And if not, the packet is sent in the direction of AP1. This is repeated until all anchor points are deleted from the anchored path. Then the packet is sent in the direction of the final destination by using GPF as described in the previous section. Figure 4.6 presents AGPF in pseudocode. Figure 4.7 illustrates how AGPF works with an example.

If the anchors are correctly set, then there is a high probability that the packet will arrive at the destination. A good anchored path directs packets along regions with good terminode connectivity. Occasionally, when there is a hole in the terminode distribution between two anchors,

X has packet p to forward to destination D via anchored path AP

$AP := p.anchored\_path$, $A1$:= $AP.get\_anchor$ //get the first anchor from the anchored path
if ($dist(X, A1) > X.transmission\_range$)
    transmit($p, GPF(A1).next\_hop$) //X uses GPF in the direction of $A1$
else $AP := AP.delete(A1)$
    if ($AP \neq \emptyset$)
        $A2$:= $AP.get\_anchor$ //get the next anchor from the anchored path
        transmit($p, GPF(A2).next\_hop$) // X uses GPF in the direction of $A2$
    else transmit($p, GPF(LDA_D).next\_hop$ // X uses GPF in the direction of D

Figure 4.6: Anchored Geodesic Packet Forwarding (AGPF) algorithm (pseudocode)

routing around the perimeter of a hole is used. We can also imagine situations when anchored path is not correctly set. Then, it may happen that there is not a greedy path from one anchor to the next another. Then, the packet may be forwarded in perimeter mode in order to come close to the anchor to be reached. During this operation, if there is a large region without terminodes in between two anchors, the packet may be lost due to the time-to-live (TTL) field expiration.

4.8 How to expedite termination of TRR

As it is described in the previous sections, TRR is the method that uses the location information in order to forward the packet as close as possible to the destination location (LDA_D), which is stamped in the packet by the source. TRR is used until some intermediate node finds that the destination can be reached by means of TLR. In this case the "use TLR" bit in the packet header is set to one. Thereon, only TLR will be only used for packet forwarding.

However, if the accuracy of location management is not sufficient, or if the packet has been delayed (due to congestion or bad paths), the "use TLR" bit may never be set. Then, the packet may start circulating around LDA_D: it is forwarded via nodes that are close to LDA_D, but the packet does not reach the destination because D has moved considerably from LDA_D and no node in vicinity of LDA_D contains anymore D in their TLR-reachable area. Finally, the packet is dropped due to the time-to-live field (TTL) expiration.

Our approach is to discover such situations and to prevent a long lifetime of circulating packets.

A node X detects the case of packet circulation if X finds that LDA_D is within its transmission range (distance(LDA_D, LDA_X) <transmission_range$_X$), and the destination is not TLR-

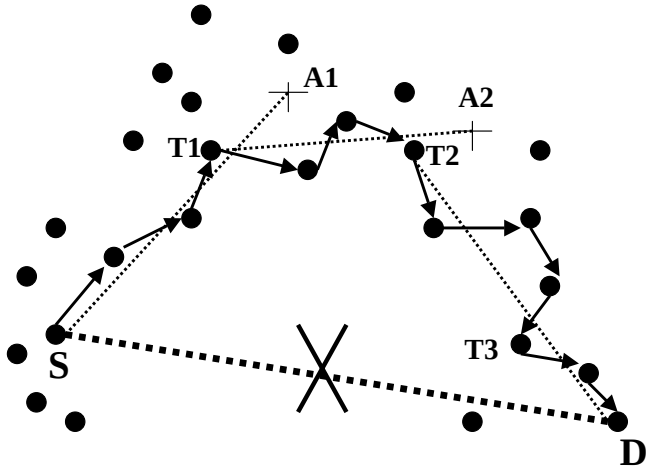


Figure 4.7: The figure presents how AGPF works when a terminode with EUI_S has some data to send to a terminode with EUI_D, and there is no connectivity along the shortest line from S to D. S has a path to D given by a list of geographic locations called anchors: $\{A1, A2\}$. First, GPF in the direction of $A1$ is used. After some hops the packet arrives at a terminode $T1$ that finds that $A1$ falls within its transmission range. At $T1$, the packet is forwarded by using GPF in the direction of $A2$. Second, when the packet comes to $T2$, that is close to $A2$, it starts sending the packet towards D. Last, when the packet comes to $T3$ it finds that D is TLR-reachable and forwards the packet to D by means of TLR.

reachable.

We propose two possible actions to solve the problem of packets that continue to circulate due to location inaccuracy. The first approach is to limit the lifetime of circulating packets. The second approach is to control flooding in the region where the destination is expected to be. Below we present the two approaches in more details.

Limited lifetime of circulating packets If X detects a circulating packet, X limits the lifetime of such a packet. In order to do so, X sets inside the packet the new value of TTL equal to $min(term\_trr, TTL)$. $term\_trr$ is a fixed value, which indicates that a loop due to destination location inaccuracy is always limited to $term\_trr$ hops[1]. After the packet has lived for $term\_trr$ hops without being delivered to D, it is dropped.

Restricted Local Flooding (RLF) helps in the case of location inaccuracy Restricted Local Flooding (RLF) controls the flooding of packets, and works as follows.

[1]In our current implementation of TRR $term\_trr$ is equal to 3.

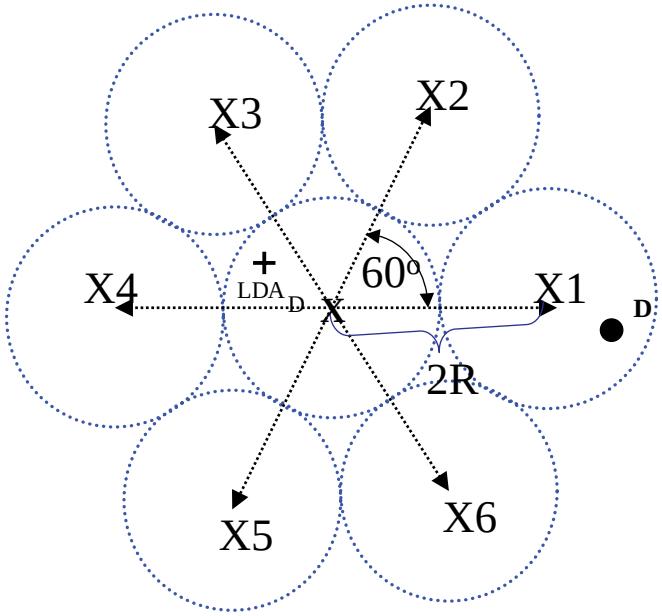


Figure 4.8: Node X has a packet for D and finds LDA_D falls within its transmission range, but D is not TLR-reachable. X performs Restricted Local Flooding (RLF) by sending six copies of the packet towards six different geographic points around X.

Again, let's say that the packet is received at node X, which finds that LDA_D falls in its transmission range, but the destination is not TLR-reachable. Then, X moves to the RLF mode.

RLF consists in sending six copies of the packet in different directions around the sending node (X). In this way, these copies are sent in the area around X, where the destination is expected to be.

Local flooding is restricted because it does not use broadcasting like common flooding, and because duplicate packets are dropped after a certain number of hops if not arrived at the destination. If instead of RLF the common flooding were used, then it would be necessary to control the flooding on a per packet basis. In order to avoid the redundant transmissions of the same packet, it would be necessary that intermediate nodes keep track of the packets that they have already seen. All this is not needed in the case of RLF because packet duplicates are forwarded in the same way as all other packets.

Within each copy, X sets the rlf bit in the packet header to one, thus denoting that the packet is in the RLF mode. X sends each copy by using GPF in the direction of one of the six geographic points around X. In Figure 4.8 these geographic positions are denoted as Xi, $i \in 1..6$. Within the i^{th} packet, the destination LDA field in the packet header is set to Xi. However, the EUI field is not changed (that is EUI_D). X1 through X6 thus present virtual destination positions.

All points $X1$ to $X6$ are at the same distance from X, which is equal to twice the transmission range of X. It can be seen from Figure 4.8, that with circles around each of six points Xi (whose radii is equal to the node transmission range), we cover the region equal to twice the transmission range. If the destination is within this region it is very probable that it receives at least one copy of the packet.

The TTL field in each copy is set to $term\_rlf$, which is a small number[2]. In this way, we constrain the lifetime of a copy to $term\_rlf$ hops. Packets where RLF is started (rlf bit is set to one) are forwarded towards one of geographic positions Xi using GPF. There are three possible situations with packets whose rlf bit is set to one.

1. In the first case, the packet is delivered to the destination by some intermediate node that finds D in its TLR-reachable area.

2. In the second case, the packet has been flooded but D is not reached and therefore the packet is dropped due to TTL expiration.

3. The third case occurs when some intermediate node N, finds X_i (Xi is written in the destination LDA field inside the packet header) in its transmission range, but the destination is not TLR-reachable, and therefore N should expedite a termination of TRR. In this case, because the packet has rlf bit set to one, N drops the packet. In this way we avoid restricted local flooding of the packet, which is itself created after the action of RLF.

RLF is valuable in the case when the accuracy of location information is low because it increases the geographic area where it is expected to find the destination. It is also possible to increase the region where RLF is applied. This can be done by taking points Xi further away from node X, and sending more packet copies. However, the drawback is the increased overhead due to packets that are duplicated and forwarded in the network when none of them reaches the destination.

4.9 Anchored Path Discovery

In this section we present two methods for path discovery, namely Friend Assisted Path Discovery (FAPD) and Geographic Maps-based Path Discovery (GMPD). The two schemes are complementary and can coexist. The first one, FAPD, assumes a common protocol in all nodes and a high degree of cooperation among nodes for providing paths. It is a social oriented path

[2]In our current implementation of TRR $term\_rlf$ is equal to 4

discovery scheme. The second one, GMPD, needs to have or to build a summarized view of the network topology, but does not require explicit cooperation of nodes for acquiring paths.

4.9.1 Friend Assisted Path Discovery (FAPD)

FAPD is a default method for obtaining anchored paths.

FAPD is based on the concept of small world graphs[85]. Small world graphs are very large graphs that tend to be sparse, clustered, and have a small diameter. The small-world phenomenon was inaugurated as an area of experimental study in social science through the work of Stanley Milgram in the 60's. These experiments have shown that the acquintanceship graph connecting the entire human population has a diameter of six or less; the small world phenomenon allows people to speak of the "six-degrees of separation".

We view a terminode network as a large graph, with edges representing the "friend relationship". B is a *friend* of A if (1) A evaluates that it has a good path to B and (2) A decides to keep B in its list of friends. A may have a good path to B because A can reach B by applying TLR, or by GPF, or because A managed to maintain one or several anchored paths to B that work well. The value of a path is given in terms of congestion feedback information such as packet loss and delay. More about path evaluation is given in Section 4.12.

Every terminode has a knowledge of a number of terminodes in its TLR-reachable region (local friends); this makes a graph highly clustered. In addition, every terminode has a number of remote friends to which it maintains a good path(s). We conjecture that this graph has the properties of a small world graph. In a small world graph, roughly speaking, any two vertices are likely to be connected through a short sequence of intermediate vertices. This means that any two terminodes are likely to be connected with a small number of intermediate friends.

With FADP, each terminode keeps the list of its friends with the following information: location of friend, path(s) to friend and potentially some information about the quality of path(s).

FAPD is composed by two elements: *Friends Assisted Path Discovery Protocol (FAPDP)* and *Friends Management (FM)*.

Friends Assisted Path Discovery Protocol (FAPDP)

$FAPDP$ is a distributed method for finding an anchored path between two terminodes in a terminode network. The concept of FAPDP can be summarize as follows. When a source S needs to discover a path to destination D, it requests assistance from some friend, let's say F. F can already have the path to D, or F tries to find the path (perhaps with the help of its own

```
if (S has a friend F1 where dist(F1,D)<dist(S,D) )
   {S sets "F" bit in the packet header; send a packet to F1;}
else if (S has a friend F3 such that dist(S, F3) < max_dist )
      {S sets "F" bit in the packet header;
       tabu_index=1; min_dist=dist(S,D); //start tabu mode
            send the packet to F3;}
else apply geodesic packet forwarding (GPF) to D;
```

Figure 4.9: Friend Assisted Path Discovery Protocol at the source

friends).

Figures 4.9 and 4.10 present FAPDP in pseudocode at the source and at an intermediate friend. In the following we describe the operation of FAPDP.

Source S, which has some data to send to D, has some friends that are closer to D than S itself, it selects friend $F1$ that is closest to D, and starts FAPDP with $F1$. S sends the data packet to $F1$ according to the existing path that S maintains to $F1$ because $F1$ is a friend of S. S sets, within the data packet header, the "F" bit[3]. This denotes that the corresponding packet is a *path discovery packet*.

When $F1$ receives this packet it recognizes the packet as a request of a path to D. The $fapd\_anchored\_path$ field inside the path discovery packet progressively contains anchor points from S to D. If S has an anchored path to $F1$, S simply puts anchors of this path in the $fapd\_anchored\_path$ field (S sends data to $F1$ with AGPF). Otherwise, S leaves this field empty (in this case S sends to $F1$ with GPF). Upon reception of the path discovery packet, $F1$ puts its geographic location inside $fapd\_anchored\_path$ field as one anchor. If $F1$ has an anchored path to D, $F1$ appends this path into the $fapd\_anchored\_path$ field and sends the packet to D by AGPF. If $F1$ does not have a path to D, it recursively uses FAPDP. In this case, $F1$ checks if it has a friend $F2$ closer to D, and then it performs the same steps as S. This is repeated until the packet is received by some intermediate node that finds D to be TLR-reachable and it forwards the packet to D by TLR.

The first example of FAPDP, presented in Figure 4.11, shows the case where the path from S to D is found by using three intermediate friends.

However, there are situations where the source or an intermediate friend does not have a friend closer to the destination. In some topologies with obstacles, at some point, going in the

[3]the "F" bit is not reset before reaching D

```
F1 is intended receiver of a path discovery packet ("F"bit = 1 ): S needs a path to D
if (F1 == D) {send path reply with fapd_anchored_path to S;}
else if (F1 has a path to D)
  append this path in fapd_anchored_path and send the packet to D;
else if (tabu_index > 0 ) //packet in tabu mode
   {
    if ( F1 has a friend F2 where dist(F2, D) < min_dist)
      {tabu_index=0; send the packet to F2}
    else if (tabu_index < 2 and F1 has a friend F3 such that dist(F1, F3) < max_dist )
         { tabu_index++; send a packet to F3}
    else // tabu_index reached the maximum value
       {send a packet to D by geodesic packet forwarding}
   }
else //packet not in tabu mode
 {
  if (F1 has a friend F2 where dist(F2,D)<dist(F1,D) )
   send a packet to F2;
  else if (F1 has a friend F3 such that dist(F1, F3) < max_dist )
      {tabu_index=1; min_dist=dist(F1,D); send a packet to F3}// start tabu mode
  else apply geodesic packet forwarding (GPF) to D;
}
```

Figure 4.10: Friend Assisted Path Discovery Protocol at the intermediate friend and at the destination

direction opposite from the destination may be the only way to reach the destination. FAPDP permits that some terminode T (the source or an intermediate friend) sends a path discovery packet to a friend even though the packet is not getting closer to the destination. However such a friend must not be distant from T more than distance max_dist[4]. Here is where the "tabu" mode of FAPDP starts. With the tabu mode mechanism, intermediate friends can send the packet in a direction opposite to D for a limited number of times. Our method is inspired by the Tabu Search heuristic ([25], [28]). Tabu Search can be defined as a general heuristic in which a local search procedure is applied at each step of the general iterative process. It could be superimposed on other heuristics to prevent those being trapped in a local minimum. We use the tabu mechanism in order to get out of a local minimum that can happen at some node that does not have a friend closer to the destination. Then, with the tabu mechanism, we try the opposite direction (non-

[4]we use max_dist equal to five times the transmission range of a terminode

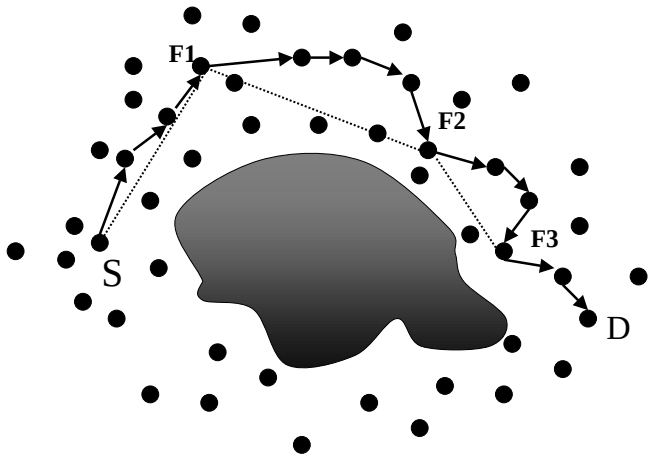


Figure 4.11: Figure presents how FAPDP works when source S, has a friend $F1$ that is closer to D than S. S sends data packet to $F1$ and sets the "F" bit in the packet header in order to denote that this is a "path discovery packet". Upon reception of the path discovery packet, $F1$ puts LDA_{F1} inside the $fapd\_anchored\_path$ field of the path discovery packet as one anchor. In this example $F1$ does not have path to D, but has a friend $F2$ whose distance to D is smaller than the distance from $F1$ to D. $F1$ sends path discovery packet to $F2$. In a similar way, $F2$ sends the packet to its friend $F3$. Once $F3$ receives the packet, it finds out that D is TLR-reachable and $F3$ forwards the packet to D by TLR. When D receives the packet with set "F" bit, it should send back to S a "path reply" control packet with the acquired anchored path from S to D. Assuming that the path from S to $F1$, from $F1$ to $F2$ and from $F2$ to $F3$ does not contain any anchors, the anchored path from S to D is thus a list of anchors $(LDA_{F1}, LDA_{F2}, LDA_{F3})$.

improving move) from the destination with the aim to finally get out of a local minimum and further approach towards the destination. In order to avoid cycling, in FAPDP we limit the number of consecutive non-improving moves.

The tabu mode is denoted at T by setting the $tabu\_index$ field inside the packet to 1 (default value of $tabu\_index$ is 0).

The tabu mode mechanism uses a field called min_dist, where the terminode that started the tabu mode puts its distance to the destination. When an intermediate friend $F1$ receives the path discovery packet, which is in tabu mode, it first checks if it has a friend whose distance to D is smaller than min_dist. If this is the case, the packet is sent to such a friend, and $tabu\_index$ is reset to 0. Otherwise, $F1$ may forward the packet to its friend $F2$ whose distance to D is more than min_dist and $F2$ increments $tabu\_index$. In FAPDP, the number of times that the packet is forwarded to a friend that is further from D than min_dist is limited to two (i.e, the value of $tabu\_index$ must not be larger than two). Tabu mode mechanism stops either because a friend that is a distance from D less than min_dist is found, or because $tabu\_index$ is equal to 2. In

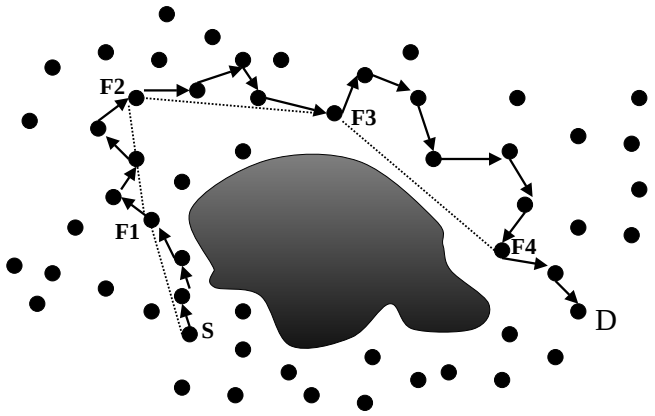


Figure 4.12: Figure presents how FAPDP works when source S does not have a friend that is closer to D than itself. S contacts its friend $F1$ that is farther from D in geometrical distance than S is, but such that $dist(S, F1) < max\_dist$. As in the previous example, S sends data packet to $F1$ with "F" bit set. In addition S sets the $tabu\_index$ field to 1 and thus starts the tabu mode of FAPDP. S puts $dist(S, D)$ within min_dist field. Upon reception of the path discovery packet, $F1$ finds out that it does not have a friend whose distance to D is smaller than min_dist. $F1$ forwards the path discovery packet to its friend $F2$ (that is in the opposite direction from D) where $dist(F1, F2) < max\_dist$, and sets $tabu\_index$ to 2. Upon reception of the packet, $F2$ checks that $tabu\_index$ is equal to its maximum value, and $F2$ cannot forward the packet to its friend that does not reduce the distance min_dist. In our example, $F2$ has a friend $F3$ whose distance to D is smaller than min_dist and forwards the packet to it. At $F3$, $tabu\_index$ is reset to 0. This means that FAPDP is not longer in tabu mode. From $F3$ packet is forwarded to its friend $F4$ and from there to D by using the TLR protocol. The anchored path from S to D is thus a list of anchors $(LDA_{F1}, LDA_{F2}, LDA_{F3}, LDA_{F4})$

the second case the packet is forwarded directly to D by GPF.

Finally, when D receives the packet with the "F" bit equal to one, D must send back to S a "path reply" control packet with the acquired anchored path from S to D. This packet is sent to S by reverting the anchored path and applying AGPF. Once S receives from D a packet with the anchored path, S stores this path in its route cache.

If S does not receive a anchored path within some time, or if S wants more paths to D, S starts FAPDP with some other friend.

The second example of FAPDP, presented in Figure 4.12, illustrates the tabu mode of FAPDP.

Friends Management

Friends Management (FM) is a set of procedures for selecting, monitoring and evaluating friends. For each node, FM maintains a (fixed-size) set of nodes: the *list of friends*. The list of friends

contains the nodes that are contacted with $FAPDP$ for discovering paths. Friends Management consists of the following components: *Friends Monitoring*, *Friends Evaluation*, *Potential Friends Discovery* and *Friends Selection.*

Friends Monitoring and Friends Evaluation

Friends are periodically evaluated in order to assure the consistency of the information on current friends and for testing the validity of these friends. We assume that some form of location tracking is active between friends. The *Friends Monitoring* component of FM keeps under control, for a node A, a set of parameters for each friend F_i of A.

A series of parameters are used to evaluate friends. These are:

1. *Value of path(s) to friend* F_i: A may evaluate that the path to its friend F_i, that worked well in the past, deteriorated. We talk about the path evaluation in Section 4.12.2.

2. *Location of friend* F_i *and the average distance to* F_i: F_i may have moved considerably from the location where is was at the time when it was included in $A's$ list of friends.

3. *The number of times friend* F_i *was contacted to provide a path and the number of paths that are found with the help of friend* F_i: A may contact F_i in FAPDP to learn the path to the destination, but the path is never returned back to A.

A terminode evaluates a friend as bad if any of the following is true: path to the friend deteriorates, or, the friend has moved considerably from the location where it was when it has been selected to be a friend, or, a friend was contacted several times in FAPDP, but the path was never acquired.

Based on these parameters, the *Friends Evaluation* component periodically evaluates whether it is beneficial to keep a node in the list of friends, or it is better to discard it.

Friends with bad evaluation results are discarded from a friends list. At run-time, initial friends disappear- very likely, in order to be substituted by more valid friends. If the number of friends that remain in a list after evaluation is low, new *potential friends* can be obtained with the *Potential Friends Discovery* component.

FM is critical in the initial phase (bootstrapping). When a node bootstraps, it does not have any information on (possible) friends. Then, the *Potential Friends Discovery* component is invoked by a node, with the aim to learn from other nodes information about some potential friends. Potential friends are also subject to the *Friends Selection* component. A number of friends are selected at random from the list of potential friends. Below, we show that the friend

selection should take into account geographic positions of potential friends in order to build a friendship graph with small world graph properties.

The next two sections describe in more detail the *Potential Friends Discovery* component and *Friend Selection* that acts on a list of potential friends.

Potential Friends Discovery

Terminode T can have frequent communications with some other terminodes (e.g., for the personal, business, or economical interest). These terminodes can be directly selected as friends, because it is of T's interest to maintain constantly path to them.

However, in general, a terminode could have a small number of terminodes that it contacts frequently. Therefore, there should be a way for terminodes to select new friends. This is the task of the *Potential Friends Discovery* and *Friends Selection* components.

With the Potential Friends Discovery component, node T receives the information on some possible friends, from other nodes in the network. At that point, T has to choose the ones that it will use as friends for the next period of time. This is performed during the Friends Selection phase, which we describe below. This applies both at the bootstrap phase, and periodically, upon request from the friend management component.

As it is already described in Section 4.4, terminodes periodically send HELLO messages, for the purpose of building the TLR routing tables. In this process, terminodes can learn about EUIs and LDAs of the one-hop and the two-hop distant nodes. Given that this information is periodically maintained, a node always has information about close nodes which can be considered as *close potential friends*.

Potential friends that are further than two hops (i.e. the node does not maintain information about their EUIs and the LDAs by means of HELLO messages) are called *remote potential friends*. One way for a node T to learn about remote potential friends is to extract this information from its previous communications. However, to avoid situations where this implicit discovery would not perform properly (e.g. after a long IDLE or OFF period), this component includes a protocol that enables a node to explicitly discover remote potential friends. In this scheme, each node T sends the $get\_friends\_request$ message towards four geographic points (GP1, GP2, GP3 and GP4). These points are randomly selected as four points in orthogonal directions at four times the transmission range of T. Once a node Y on the way towards a point FPi finds that FPi falls in its transmission range, it stops forwarding the $get\_friends\_request$ message. Then Y sends back the $get\_friends\_reply$ message to T, which contains the list of friends of Y. If this table is empty, Y puts itself in the content field of the message. When node T eventually

receives the $get\_friends\_reply$ message from the node Y, it combines the received information with the current one in its list of friends.

After T acquires a list of potential friends, it applies the *Friend Selection* component to select a certain number of friends that it includes in the friends list. We do not define, in this context, how large is the number of friends that a node maintains in its list of friends. This is the matter of ongoing work.

Friends Selection: On How to Build a Small World Graph of Friends

New friends are selected from a list of potential friends by the *Friends Selection* component.

As we said before, FAPD works on the network of friends: the idea behind it is that such a network has the properties of a small world graph where only a small number of intermediate friends is needed in order to connect any two nodes in the network.

What we want to achieve is that terminodes select their friends in the way that the resulting friendship graph has properties of a small world graph. In the friendship graph vertices correspond to terminodes, and there is an edge between nodes i and j if i keeps j in its list of friends.

The key to generate the small-world phenomenon is the presence of a *small fraction* of long-range edges, which connect otherwise distant parts of the graph, while most edges remain $local$, thus contributing to the high clustering property of the graph.

Our strategy is to consider geographic positions of nodes when building friends connections. We distinguish two types of friendship connections:

- *short-range friendship connections (local)* correspond to one hop distant terminodes (physical neighbours). These local friendship connections aim to make a friendship graph clustered.

- *long-range friendship connections (shortcuts)*: correspond to "logical" connection to terminodes that are more than one hop distant. Each node chooses a small number of them. A shortcut is represented in a friendship graph as one edge.

In order to determine its shortcuts a node takes into consideration distances from other nodes in the graph. A node chooses with higher probability friends that are closer to it. However, there is always some probability that it will choose some distant friend. Our strategy for choosing long-range contacts is inspired by Kleingberg's paper[43].

Kleingberg in [43] considers a two-dimensional square lattice, where each node is joined to its four nearest neighbours. Then, for each vertex one shortcut is added, but not purely at

random. For each vertex, all the possible destinations of a shortcut link are assigned a rank based on their lattice distance from the source vertex. The probability of choosing a vertex at distance d is proportional to d^{-r}, where r is an additional parameter of the model. In the case when $r = 0$, shortcuts are chosen with uniform probability. Then with a high probability, there are paths between every pair of nodes and these paths are bounded by a polynomial in $log(n)$, exponentially smaller than the total number of nodes n. However, there is no way for a decentralized algorithm to find these paths. When r is large, then only close nodes have a chance to be connected with a shortcut. The key value for r turns out to be 2. When $r = 2$, it is shown that the resulting graph is a SWG and there is a distributed algorithm for finding short paths between any two vertices (paths are exponentially smaller than the total number of nodes). This algorithm is $greedy$. In Kleinberg's paper [43] this algorithm is as follows: in order to find a path from vertex S to vertex D, S lists all edges that come out of it, and chooses the one that connects S to the vertex that is closest to D, as measured by lattice distance; then repeat the same procedure until D is reached.

Inspired by the Kleinberg's results, we propose the following. The probability that node X_i selects node X_j as its long-distance friend from the list of potential friends ($X_k, k \in 1..n, k \neq i$) is given with the formula:

$$p(X_i, X_j) = \frac{1/dist^2(X_i, X_j)}{\sum_{k=1,k\neq i}^{n} 1/dist^2(X_i, X_k)} \tag{4.1}$$

In this formula, we denote with $dist$ the geographical distance between two nodes. The formula says that the probability for node X_i to choose friend X_j is proportional to $dist(X_i, X_j)^{-r}$, with r=2. Shortcuts are thus selected at random and are not necessarily bidirectional. X_i may have X_j as a friend, but X_j may not have X_i as a friend.

We use simulations to verify our strategy for selection of long-range friendship connections. We performed simulations with the following assumptions:

- Nodes in the network are distributed as a two-dimensional Poisson point process with density λ.
- All nodes have the same the transmission range R.
- All nodes have a knowledge of identities and locations of all other nodes.

Initially, a friendship graph contains only short-range connections. There is a short-range connection between X_i and X_j, if dist(X_i,X_j)$\leq R$.

Then, every node selects a number of shortcuts from its list of potential friends. We performed simulations where this number is equal to one or two. In our simulations, a node has in a list of potential friends the whole set of nodes except nodes with whom short-range connections are already established.

The algorithm that node X_i uses to select its friends consists of the following three steps:

- Step 1: If node X_i keeps n nodes in its list of potential friends, interval $[0, 1]$ is divided into n intervals. The length of j^{th} interval is equal to $p(X_i, X_j)$, given by Equation (4.1).

- Step 2: For each friend to be selected, a random trial is performed: a uniform random deviate r in interval $[0, 1]$ is generated. If r falls in the j^{th} interval, X_j becomes a friend of X_i.

- Step 3: The same procedure is repeated for each friend that X_i selects.

The friendship graph is built when all nodes select their friends. Then, we find the characteristic path length of the graph. The *characteristic path length* (CPL) of a graph is the $median$ of the $means$ of the shortest path length connecting each vertex to all other vertices. CPL in a way presents the typical shortest path length between every vertex and every other vertex. CPL is also used by Watts in [85] as a metric to verify whether a graph is a small world graph. A small world graph has a small CPL.

The purposes of our simulations are twofold. First, we want to verify by simulations that adding a small number of shortcuts in a friendship graph reduces the CPL of the graph. Second, we want to verify that a greedy algorithm for finding paths succeeds in finding short paths. With the greedy algorithm, the source and every intermediate node forward the packet to their short or long-range friend that is closest to the destination. FAPDP is basically a greedy algorithm, and uses the "tabu" mode of operation only when the greedy forwarding is not possible. In our simulation we also calculate CPL, where instead of shortest paths between nodes, we use paths lengths obtained by the greedy algorithm.

Simulation results are given in Figure 4.13, averaged over ten realizations of random graphs for a given number of nodes. In our simulations, transmission range is (R) is 250 meters. Node density is such that every node has an average of ten neighbours (short-range friendship connections). As we increase the number of nodes, we increase the simulation area, but we keep the same density of nodes. The chosen density ensures that the greedy algorithm succeeds in finding most of the paths. We verified that for less than 5% of pairs of nodes, the greedy algorithm is not able to find a path (see Figure 3.6).

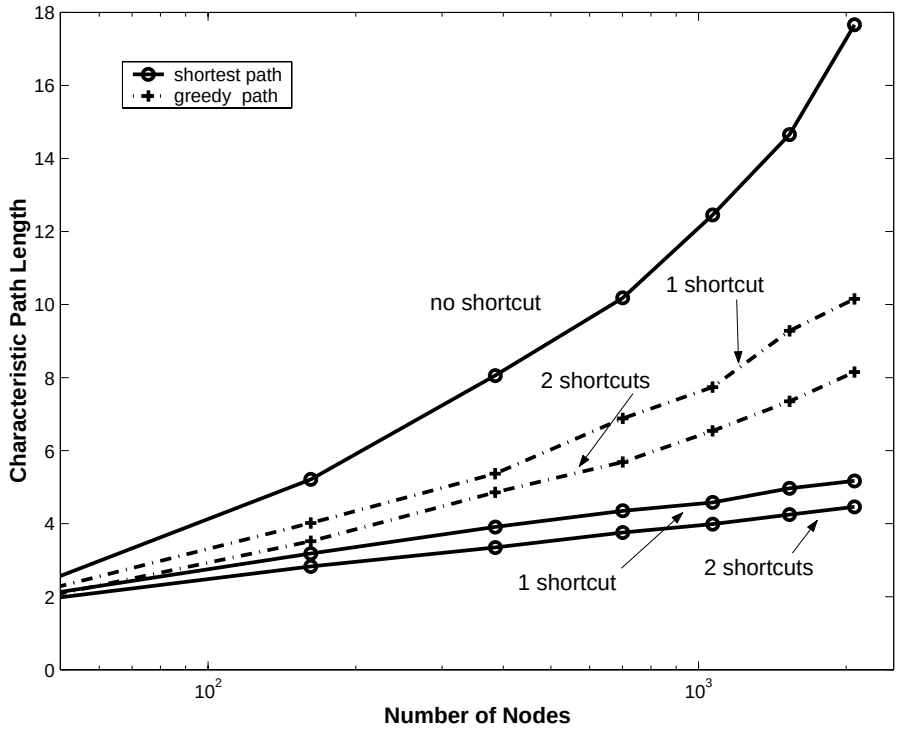


Figure 4.13: Figure presents the characteristic path length (CPL) of friendship graph for different number of nodes. Adding a small number of long-range shortcut edges in the graph reduces CPL.

Our simulations have shown the following. First, CPL of the friendship graph exhibits logarithmic length scaling with respect to number of nodes in the graph (see Figure 4.13). This is a property of a small world graph. A small number of long-range connections (e.g., 1 or 2) are enough to reduce CPL considerably from the case when shortcuts are not used. Second, the greedy algorithm for finding paths succeeds in finding paths whose length is close to shortest paths.

In order to perform FAPDP, a terminode uses its local and long-range friendship connections. As described in 4.9.1, a node first contacts its long-range friends (if there are such friends) in order for the path discovery packet to come as close as possible to the destination location. However, if a node does not have a long-range friend, it sends the path discovery packet to its local friend (one hop away). Our goal is to get a good anchored path with as small number of anchors as possible. Therefore, in the scenario where there is a sequence of local friends that are contacted, one after the other, it is not necessary that each local friend puts its location as one anchor in the accumulated anchored path. We limit the maximum number of consecutive

local friends to six, before the next local friend's location becomes an anchor in the accumulated anchor path. This is always the case, unless a local friend has a long-range friend. In this case, a local friend's location becomes an anchor before it forwards the packet to its long-range friend.

4.9.2 Geographic Maps-based Path Discovery (GMPD)

GMPD is another method for anchored path discovery, which assume that the network topology is known to all nodes in the network.

A terminode network is a large area mobile ad hoc network. We believe that a good model of a large mobile network does not assume that nodes are uniformly distributed in the network. In order to model a terminode network, we identify the areas with a higher node density, which we call $towns$. Two towns are interconnected by all the nodes in between them (we call it a $highway$). If two towns are interconnected with a highway, there is a high probability that there are terminodes to ensure connectivity from one town to another. GMPD assumes that each terminode has a summarized geographic view of the network. Each terminode has a knowledge of a map of towns. A map defines the network topology: it defines town areas and reports the existence of highways between towns. As a first attempt, we model a town area as a square centered in a geographic center. For each town, a map gives the position of its center and the size of the square area. One example of a map of a terminode network is presented in Figure 4.14. A map of the network can be presented as a graph with nodes corresponding to towns and edges corresponding to highways. Macroscopically, the graph of towns does not change frequently.

GMPD with a given map of towns works as follows:

- Source S determines from its own location LDA_S the town area (ST) in which S is situated (or, the nearest town to LDA_S if it is not in the town area). In addition, since S knows the position of destination D (LDA_D), it can determine from the LDA_D the town area DT where D is situated (or, the nearest town to LDA_D if it is not in the town area).

- Then, S accesses the network map in order to find the anchored path from S to D. We call this operation a *map lookup*. An anchored path is the list of the geographical points: the points correspond to centers of the towns that the packet has to visit from ST in order to reach DT. One possible realization of the map lookup operation is to find a list of towns that are on the shortest path from ST to DT in the graph of towns; the length of a path can be given either as the number of towns between ST and DT, or the length of the topological (Euclidean) shortest path connecting ST and DT in a graph of towns.

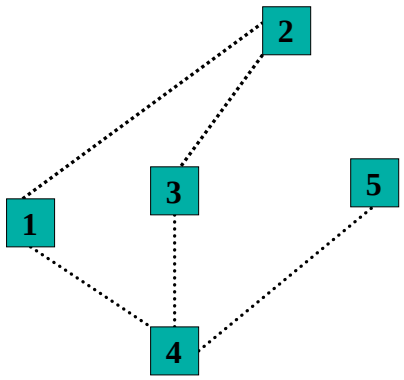


Figure 4.14: Figure presents one example of a map of a terminode network. Five town areas (A, B, C, D and E) are presented with shaded squares. A highway between two towns is presented with a line between two town areas.

In Section 4.12, we explain that the source should distribute data packets to the destination along multiple anchored paths, for the reasons of better load balancing and the path failure protection. Having the map of towns, the source can perform several map lookup operations in order to find several disjoint paths to the destination. Then these paths can be used in parallel for packet forwarding.

GMPD with no initial summarized view of the network

Here, we still assume the model of a network based on towns and highways, however, nodes in the network have a constrained view of a network.

A terminode initially does not have the knowledge of a map of towns. The information that a terminode has is the following: 1) if a terminode is within a town area, it knows about neighboring towns areas and town centers with which its current town is connected by highways; 2) if a terminode is on the highway, it knows towns that this highway connects. We assume that a graph of towns is planar. As above, we assume that there is a mapping between the location of a terminode and the corresponding town. Thus, source S can determine from LDA_S the town area (ST) in which S is situated (or, the nearest town to LDA_S if it is not in the town area). In addition, S determines from the LDA_D the town area DT where D is situated (or, the nearest town to LDA_D if it is not in the town area).

Here we present the description of the path discovery algorithm with these assumptions. As before we denote that the source is in town ST and the destination is in town DT.

- if ST and DT are the same, then S does not perform path discovery; S sends the packet

to D by GPF. If ST and DT are not the same, S begins anchored path discovery. S uses a greedy method: it sends the path discovery packet towards a neighboring town NT whose center is the closest to DT. S sends the path discovery packet by using GPF towards the center of NT. As soon as the path discovery packet is received by some terminode N in NT, N adds the center of NT as one anchor in the accumulated anchored path. In addition, N adds NT in the accumulated list of towns (the list of towns is used to record towns that the path discovery packet has visited. This list is later used to simplify the path from ST to DT). If N has a path to DT, it adds this path to the the accumulated anchored path and applies this path to send the packet to D. Otherwise, N repeats the same steps as S.

- if S or an intermediate node (that has to determine the next town to which to send the path discovery packet) does not find a neighboring town closer to DT, the planar graph traversal method is used to determine the next town NT. This method, described in Section 3.3.2, is applied on a graph of towns. One example of traversal of the graph of towns is presented below.

  As soon as the path discovery packet arrives at some town that is closer to DT than the town where the planar graph traversal method is started, the greedy method resumes in order to find the next town to send the path discovery.

- the explained procedure is repeated until the packet is received by some node in DT. Then the packet is sent directly to D by GPF. When D receives the packet, it analyzes the path and filters possible loops. Then the path is sent back S.

- when S receives the path to DT it adds this path in the list of anchored paths.

We illustrate the path discovery with a localized view of the network with one example. Assume in Figure 4.14 that source S is in town 1, and destination D is in town 5. S chooses to send the path discovery packet towards the center of town 2 because the center of town 2 is closer geographically to town 5 than the center of town 1. The first node in town 2 that receives the path discovery packet puts the center of town 2 as one anchor in the accumulated anchored path. Now, because the packet cannot be forwarded closer to town 5, the planar graph traversal algorithm is used. This algorithm is applied on the planar graph of the network map. Following the right hand rule, the first edge of the graph in the direction counterclockwise from the line connecting town 2 and town 5 is the edge that connects town 2 to town 1. Therefore, the packet is sent again to town 1. The planar graph traversal is continued and from town 1 the packet is

forwarded to town 4, and then to town 5. As soon as some terminode in town 5 receives the packet it sends it directly to D by using GPF. When D receives the path discovery packet, the accumulated town list is $\{1,2,1,4,5\}$. D simplifies the list such that the same town is not visited more than once. The anchored path to be returned to S is the list of towns $\{1,4,5\}$ centers.

4.10 Estimation of Necessity of Using Anchors

How can the source estimate whether anchors are necessary in order to forward packets to the destination? In this section we address this problem. If the source and the destination are well connected along the shortest geodesic line, the basic geodesic packet forwarding (GPF) method works well. In this case packets are mostly forwarded in a greedy mode: the source and intermediate nodes find neighbours that are closer to the destination that they use as a next hop terminode. Otherwise, if the distribution of terminodes from the source to the destination is such that greedy forwarding is not possible, packets may travel along long paths in the perimeter mode. In this case, it is beneficial for the source to consume its resources to discover the anchored path to the destination and use AGPF to forward its packets.

In this section we present how source S estimates how well GPF performs in forwarding data. First, we propose that source S estimates the number of hops that the packet would take to destination D along the *greedy path*. A greedy path exists when the source and all the intermediate nodes find, among the neighbours, the next hop node that is closer to the destination. We present below the method for the estimation of the number of hops along a greedy path. This method makes use of the geographic distance from the source to the destination, and density of nodes in the network. Second, S sends explorer packets[5] that are forwarded to D, by using GPF. S learns the number of hops it took explorer packets to reach D. Then, the source compares the estimated number of hops between S and D to the number of hops it really took the explorer packets to reach the destination.

If the explorer packets have taken a much longer path than estimated, S can interpret that the direct path to D does not work well (e.g., there is an obstacle in between S and D). In this case S decides to start the path discovery algorithm to learn the anchored path to the destination.

Below we present how S finds the distribution of the number of hops it takes GPF to send data to D, based only on the knowledge of the geographic distance from S and D and the nodes density.

[5]This is similar to the $ping$ service in the fixed Internet.

Estimation of number of hops from the source to the destination We are interested in finding the number of hops from S to D when the distance from S to D is equal to d. Let's denote this as $N_{SD}(d)$.

We assume that nodes in the network are distributed as a two-dimensional Poisson point process with density λ, i.e., the probability of finding i nodes in an area of size A is equal to: $(\lambda A)^i exp(-\lambda A)/i!$, i = 0,1,2,3 ... We also assume that all nodes have an equal transmission range (R).

The average number of nodes (N) within transmission range R is then,

$$N = \lambda \pi R^2$$

At first, we find the average number of hops from S to D assuming that there is a *greedy path* that connects S and D. Let the random variable $(N_{SD}(d) \mid N_{SD}(d) < \infty)$ represent the number of hops between S and D along the greedy path that connects them. The average value of this random variable is denoted as $E[N_{SD}(d) \mid N_{SD} < \infty]$. Obviously,

$$E[N_{SD}(d) \mid N_{SD}(d) < \infty] = 1, \text{ for } d \leq R$$

For $d \geq R$, $E[N_{SD}(d) \mid N_{SD} < \infty]$ is given with the following recursive equation:

$$E[N_{SD}(d) \mid N_{SD}(d) < \infty] = 1 + \int_0^R E[N_{XD}(d-z) \mid N_{XD}(d-z) < \infty] dF_{N_{SD}}(z,d) \quad (4.2)$$

Equation (4.2) is obtained as follows: in order to reach D, S sends the packet to X because X is closest to D among all neighbours of S (for illustration see Figure 4.15. In this way the number of hops from S to D is equal to one plus the average number of hops from X to D. The progress that is made by forwarding the packet from S to X is equal to z, that is, at X the distance to D is reduced by z. At X, it remains the distance $(d - z)$ to reach D. Since the progress that can be made in one transmission is between 0 to R, the integral in Equation (4.2) calculates the average number of hops from X to D, averaged on the progress that is made from S to X.

$F_{N_{SD}}(z,d)$ in Equation (4.2) presents the conditional distribution of the progress that is made at the node where the distance to D is equal to d assuming that there exists the greedy path from S to D. $F_{N_{SD}}(z,d)$ is given by the following equation.

$$F_{N_{SD}}(z,d) = P_r(Z \leq z \mid N_{SD}(d) < \infty) = \frac{P_r(Z \leq z, N_{SD}(d) < \infty)}{P_r(N_{SD}(d) < \infty)} \tag{4.3}$$

where,

$$P_r(Z \leq z, N_{SD}(d) < \infty) = \int_0^z f(u,d) P_r(N_{XD}(d-u) < \infty) du \tag{4.4}$$

In Equation (4.3) we denoted with $f(u,d)$ the density function of the progress u made in one hop when the distance to the destination is equal to d.

From (4.3) and (4.4) we obtain,

$$\frac{dF_{N_{SD}}(z,d)}{dz} = dP_r(Z \leq z \mid N_{SD}(d) < \infty)/dz = \frac{f(z,d)P_r(N_{XD}(d-z) < \infty)}{P_r(N_{SD}(d) < \infty)} \tag{4.5}$$

From (4.2), (4.3), (4.4) and (4.5) follows,

$$E[N_{SD}(d) \mid N_{SD}(d) < \infty] = 1 + \int_0^R \frac{f(z,d)P_r(N_{XD}(d-z) < \infty)}{P_r(N_{SD}(d) < \infty)} E[N_{XD}(d-z) \mid N_{XD}(d-z) < \infty] dz \tag{4.6}$$

We can obtain the probability of existence of the greedy path from S to D by using the following recursive equation.

$$P_r(N_{SD}(d) < \infty) = \int_0^R P_r(N_{XD}(d-z) < \infty) f(z,d) dz \tag{4.7}$$

In the following we present how the density function of the progress made in one transmission ($f(z,d)$) is determined.

Let random variable Z be the progress for the transmission from node S to X. The distance from S to destination D is equal to d. We assume that the progress performed at two hops is independent, i.e., the distribution of the progress at the current node does not depend on the progress made in the previous hops. This assumption is reasonable in the case of ad hoc networks where the network topology is changing either because of node mobility or because nodes are going up and down (e.g., in sensor networks). In this case, we assume that the network topology is redrawn at every hop that receives the packet. Then the probability distribution function of Z is determined as follows. For the illustration see Figure 4.15.

$$F_Z(z,d) = P_r(Z \leq z) = P_r(\text{no nodes in } A_z) = e^{-\lambda A_z} \quad 0 < z \leq R \tag{4.8}$$

In (4.8) we denoted with A_z the excluded region without nodes and the surface of this region is equal to the sum of two surfaces P_1 and P_2, given by (4.10) and (4.11). Under the assumption

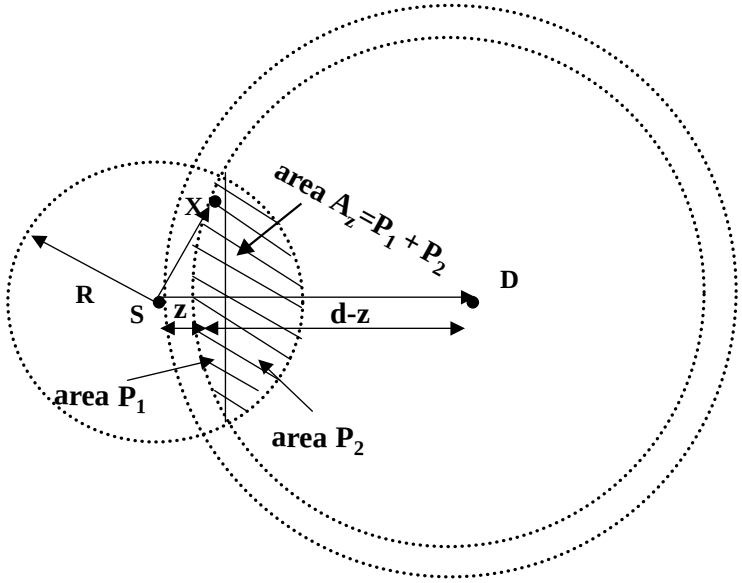


Figure 4.15: Progress in distance made in transmission from S to X is equal to z. Three circles are presented: the first centered at S and radius R, the second centered at D and radius d, and the third centered at D and radius $d - z$. The progress from S to D is less than z when there is no nodes in shaded area A_z.

that progress made at different hops is independent, the excluded region depends only on the current node distance to D, but not on other excluded areas.

$$A_z(d) = P_1(d, z) + P_2(d, z) \tag{4.9}$$

$$P_1(d, z) = R^2(arccos(a) - a\sqrt{1 - a^2}),\ a = \frac{R^2 + d^2 - (d - z)^2}{2dR} \tag{4.10}$$

$$P_2(d, z) = (d - z)^2(arccos(b) - b\sqrt{1 - b^2}),\ b = \frac{d^2 + (d - z)^2 - R^2}{2d(d - z)} \tag{4.11}$$

Then we can write the probability density function of Z as,

$$f_Z(z, d) = \frac{dF_Z(z, d)}{dz} = -\lambda e^{(P_1(d,z)+P_2(d,z))}(\frac{\partial P_1(d, z)}{\partial z} + \frac{\partial P_2(d, z)}{\partial z}) \tag{4.12}$$

where,

$$\frac{\partial P_1}{\partial z} = \frac{2(z - d)R}{d\sqrt{1 - a^2}}$$

$$\frac{\partial P_2}{\partial z} = 2(d-z)b\sqrt{1-b^2} - 2(d-z)arccos(b) - \frac{\sqrt{1-b^2}}{d}(2dz - z^2 - R^2)$$

Standard deviation (σ) of $E[N_{SD}(d) \mid N_{SD} < \infty]$ can be obtained as follows:

$$\sigma^2 = E[N_{SD}^2(d) \mid N_{SD}(d) < \infty] - E^2[N_{SD}(d) \mid N_{SD}(d) < \infty] \quad (4.13)$$

where $E[N_{SD}^2(d) \mid N_{SD}(d) < \infty]$ is obtained as follows:

$$(N_{SD}(d) \mid N_{SD}(d) < \infty) = 1 + (N_{XD}(d-z) \mid N_{XD}(d-z) < \infty) \quad (4.14)$$

$$(N_{SD}(d) \mid N_{SD}(d) < \infty)^2 = 1+2(N_{XD}(d-z) \mid N_{XD}(d-z) < \infty)+(N_{XD}^2(d-z) \mid N_{XD}(d-z) < \infty) \quad (4.15)$$

From (4.15),

$$E[N_{SD}^2(d) \mid N_{SD}(d) < \infty] = 1 + 2\int_0^R E[N_{XD}(d-z) \mid N_{XD}(d-z) < \infty]dF_{N_{SD}}(z,d) + \int_0^R E[N_{XD}^2(d-z) \mid N_{XD}(d-z) < \infty]dF_{N_{SD}}(z,d) \quad (4.16)$$

Introducing (4.2) in (4.13) we obtain:

$$E[N_{SD}^2(d) \mid N_{SD}(d) < \infty] = -1 + 2E[N_{SD}(d) \mid N_{SD}(d) < \infty] + \int_0^R E[N_{XD}^2(d-z) \mid N_{XD}(d-z) < \infty]dF_{N_{SD}}(z,d) \quad (4.17)$$

And finally standard deviation σ is given as,

$$\sigma^2 = -1 + \int_0^R E[N_{XD}^2(d-z) \mid N_{XD}(d-z) < \infty]dF_{N_{SD}}(z,d) + 2E[N_{SD}(d) \mid N_{SD}(d) < \infty] - E^2[N_{SD}(d) \mid N_{SD}(d) < \infty] \quad (4.18)$$

We can obtain the distribution of number of hops using the following recursive equation:

$$P_r(N_{SD}(d) > k \mid N_{SD} < \infty) = \int_0^R dF_{N_{SD}}(z,d) P_r(N_{XD}(d-z) > (k-1) \mid N_{XD}(d-z) < \infty) \tag{4.19}$$

We have numerically solved equations (4.2), (4.7), (4.18) and (4.19) for different distances between sources and destinations. We assume that the transmission range is the same for all nodes and is equal to 250 meters and different average numbers of nodes in the transmission range (different node densities) are considered. Figures 4.17, 4.18 and 4.19 present our results for the average number of hops, standard deviation of number of hops and hop number distribution.

Here we present the method that we used to numerically obtain our results, starting from the equations developed in this section. At first, we numerically find probabilities of existence of the greedy path, between the source and the destination for different distances between them, by using Equation (4.7). The initial values for probabilities used in this recursive equation are: $P_r(N_{SD}(d) < \infty) = 1$, for $d \leq R$. For $d \geq R$, we numerically solve the integral in (4.7) recursively using the values for probabilities that are already obtained for distances in range $(d - R, d)$.

The obtained probabilities are used as input to get the average hop count, as from (4.6). The initial conditions used in (4.6) are: $E[N_{SD}(d) \mid N_{SD} < \infty] = 1$, for for $d \leq R$.

In a similar way, we numerically get values of the standard deviation of the number of hops, given the distance between the nodes. The initial conditions used in (4.18) are: $E[N_{SD}^2(d) \mid N_{SD}(d) < \infty] = 1$, for for $d \leq R$.

Finally, the distribution of number of nodes is obtained from (4.19) where the initial values are $P_r(N_{SD}(d) > 0) = 1$, for all values of d.

Figure 4.16 presents the average hop number and standard deviation as a function of distance for different values of node densities. Figure 4.17 the presents average hop number with a 95% confidence interval for the case of the density where the average number of neighbours in the transmission range is 10.

The distribution of the number of hops is presented in Figure 4.19. In the same figure, normal distribution is presented with the mean value and variance equal to the values obtained from Equations 4.6 and 4.18. We see from Figure 4.19 that the distribution of the hop number is close to the normal distribution for various values of distances between the source and the destination. Thus, the distribution of the number of hops can be modeled by the normal distribution.

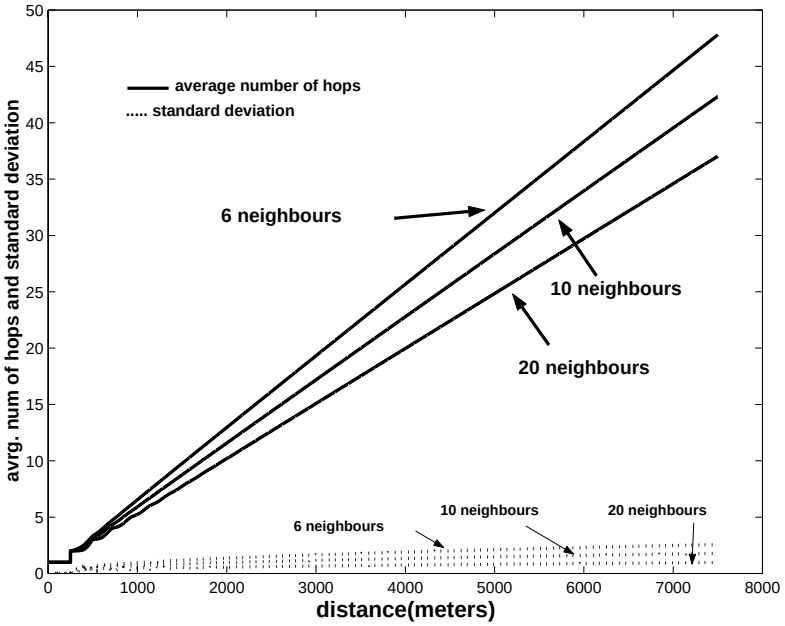


Figure 4.16: Average number of hops and standard deviation obtained numerically for different node densities

We also verified our theoretical results by simulations. We performed a number of experiments in the fixed network. Nodes in the network are randomly placed according to the Poisson distribution with the given density. In such a network, since it is fixed, the progress made in different nodes is not independent; excluded areas in Figure 4.15 are not independent. For every two nodes in the network, we found the number of hops of the greedy path that connects them, if such a path exists. The crosses in Figure 4.18 present the obtained length of the greedy path as a function of the distance between two nodes.

We can see from Figure 4.18 that number of hops obtained in experiments fall closely into a 95% confidence interval obtained theoretically. Our simulations verified that the number of hops obtained theoretically, where it is assumed that progress in different hops are independent, are close to experimental results within the fixed network where this assumption is not valid. Therefore, we conclude that obtained theoretical results will also be valid in real ad hoc networks with the mobility degree in between two extremes: fixed networks and networks where at every hop the distribution of nodes is drawn independently from previous hops.

How the obtained results can be applied in a terminodes network In order to estimate the number of hops to the destination D along a greedy path, source S should first estimate the

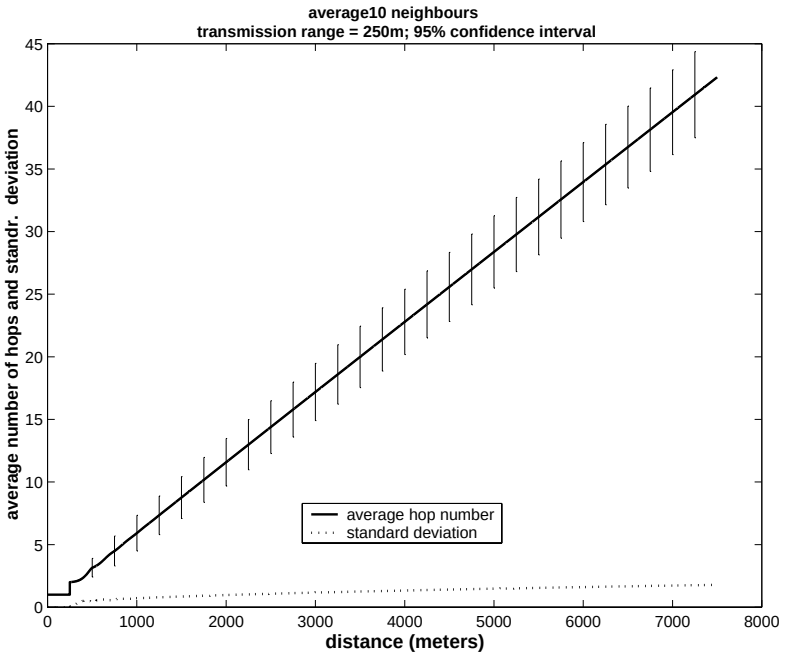


Figure 4.17: Average number of hops (95% confidence interval) and standard deviation obtained numerically for average number of neighbours equal to 10

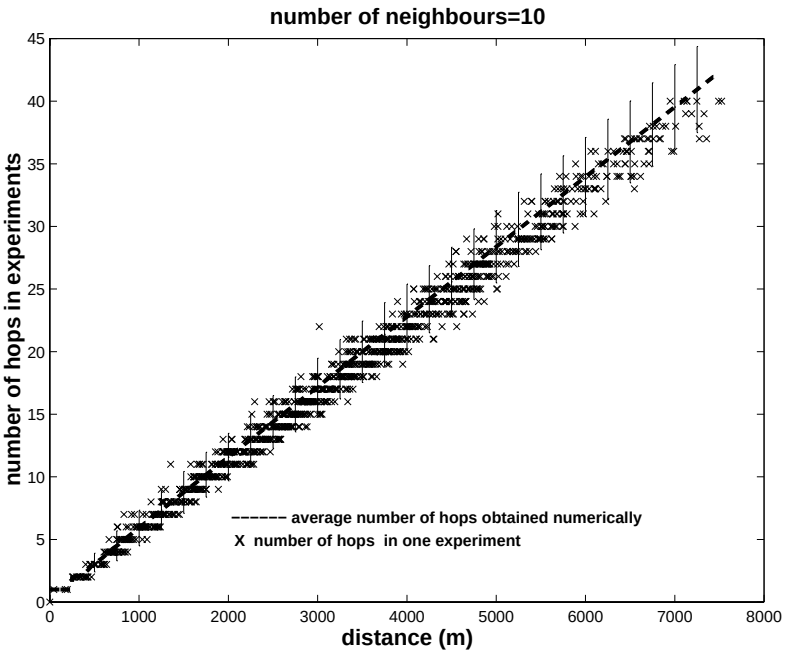


Figure 4.18: Number of hops obtained in experiments

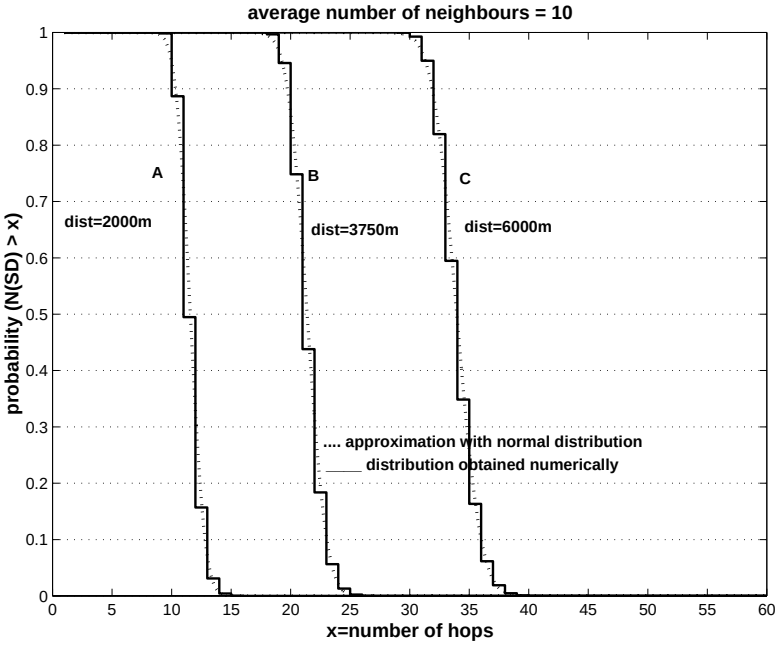


Figure 4.19: Distribution of number of hops for average number of neighbours equal to 10

density of the terminodes in a network. As the first attempt, we propose that S determines the density of nodes in its transmission range from the information in its local routing table. S assumes that the same density applies to the whole network. Knowing the geographic distance to D, S finds the distribution of the number of hops to D by applying the results that we have developed in this chapter.

Then, S sends $explorer$ packets to D that are routed using GPF. D is supposed to send back to S the response of how many hops it took the explorer packet to reach from S to D. Then, S can make conclusions about the existence of a greedy path from S to D.

For example, let's assume that S estimates that the average number of neighbours in a terminode network is ten, and the distance to D is equal to 3750m. Then from Figure 4.19 the probability that the number of hops from S to D is higher than 23 is equal to 5%. Therefore, if S learns that the explorer packets have taken more than 23 hops to reach D, S may conclude with a high probability that the greedy path from S to D does not exist. Then, S may use the FAPD method to find the anchored path that it will use to reach D.

The assumption that the node density in the network is uniform may not be true. If this is the case, the distributions of number of hops for different node densities are taken into account (when evaluated whether explorer packets have taken a greedy path or a perimeter-mode packet

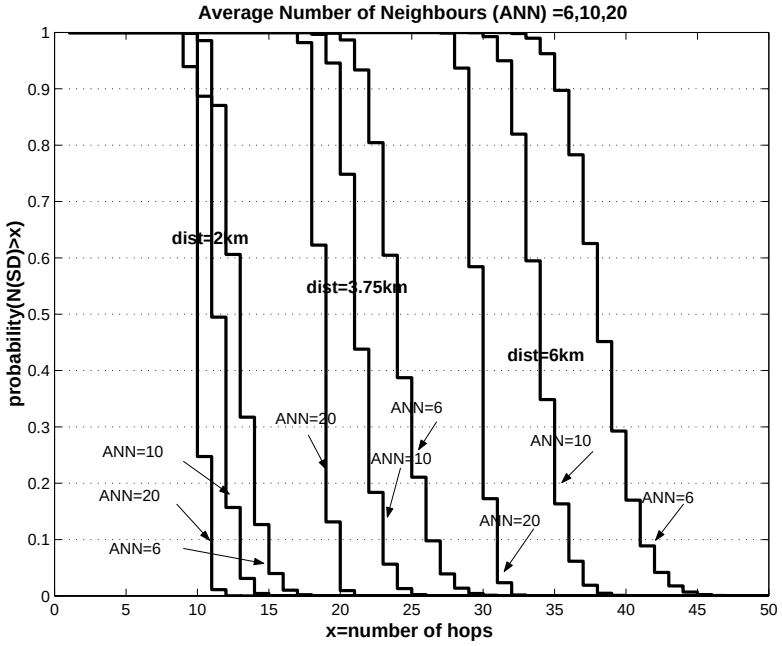


Figure 4.20: Distributions of number of hops for various node densities (average number of neighbours is 6, 10 and 20) and distances (2km, 3.75km, and 6km).

forwarding has taken place). Figure 4.20 presents distribution of number of hops for three values of the node density (average number of neighbours is 6, 10 or 20) and three different distances are taken (2km, 3.75km and 6km). We can see the larger distances are, the bigger difference is in hops distributions for different nodes densities.

We give an example for the case when the distance between the source and the destination is 6km. If the explorer packets do not take more than 45 hops to reach the destination (that corresponds to the lowest density of the average of 6 neighbours), then the source may consider that with a high probability there is the greedy path to the destination.

All the results presented in this section are obtained assuming that the whole network is populated with nodes, and there are no regions without nodes. If there is an obstacle somewhere, we propose that in addition to the number of hops it takes an explorer packet to reach the destination, the source uses the information of the number of hops that the packet has been in the perimeter mode until arrived at the destination. If the maximal perimeter length is small but the number of hops is high, there is an obstacle. The possible scenario is that the packets are going around the obstacle. Then, although a greedy path exists, anchors may be useful to find other paths that are avoiding going around not the obstacle.

Related Work Hou and Li in [31] considered routing in multihop packet radio networks. They analyzed different transmission strategies in the case whe re a transmitter uses the location information of its neighbours to find the next hop. One strategy that they presented is MFR (*Most Forward within Radius*) that we have already mentioned in Chapter 3. In MFR, a transmitting node chooses as the next hop, the node that makes the largest forward progress to the destination. The progress is defined as the distance between transmitting node and the receiving node, projected onto a line drawn from the transmitter to the final destination. In [31], the probability distribution function of progress that is made in one hop is determined, using a similar method to the one that we used to find the progress (Figure 4.15, Equation 4.8). Similar to our approach, it is assumed that progress made in different hops are independent.

However, in [31], the progress in one hop is projected on a line drawn from the current transmitter to the final destination, and is independent of the distance of the current transmitter to the destination. Therefore, the result in [31] cannot be used to find the distribution of the number of hops from the source to the destination, when MFR is used.

The result of Hou and Li[31] is used in [90] to estimate the average number of hops from the source to the destination, when MFR is used for packet forwarding. Given the distance from the source to the destination, the average number of hops is obtained by dividing the distance and the average progress made in one hop (not the function of the distance). However, in [90], the distribution of the number of hops is not determined.

4.11 Location Management in a terminode network

In this section we define the requirements of terminode routing on location management, and we give an overview of the existing location management approach that we believe can meet these requirements.

Mobility management in a terminode network is performed by the following components:

- First, TLR is able to track a destination terminode in the vicinity of a relaying terminode.
- Second, the location management enables the source to learn the location-dependent address of the destination (LDA), which is necessary for TRR. The LDA management is performed by two components. First, a *location tracking* algorithm is assumed to exist between terminodes when they have successfully established communication; this allows communicating terminodes to continuously update the corresponding LDA information. Second, a *location discovery service* is used to obtain a probable location of terminode B

(LDA_B) that A is not tracking by the previous method.

4.11.1 Terminode Routing Requirements on Location Management

The main objective of location management is to distribute the location information inside the network in a dynamic, scalable, secure, and fair way, i.e., without privileging any node or region. At the same time, in terminode routing, we are not concerned with maintaining exact location information. As already presented, terminode routing works even when a source does not maintain its exact destination location. All that is required for terminode routing is that the LDA of the destination, learned at the source, is accurate enough so that the packet eventually arrives at some terminode that finds the destination inside its TLR-reachable area.

Here, we give a rough estimation about a destination location update interval that is necessary for successful tracking of the destination. Our estimation is based on the following assumptions: (1) terminodes move with the maximum speed of 20 meters per second, which corresponds to the speed of a car; (2) a terminode applies TLR in its two-hop neighbourhood; (3) terminodes have a nominal transmission range of 250 meters (250 meters corresponds to characteristics of Lucent Technologies WaveLAN[13]), but we assume that the effective transmission range is around 100 meters. When TRR is used to forward the packet to destination D, D's location known at the source (LDA_D) is used as the direction for packet forwarding. TRR terminates if the packet is received by some intermediate node X, close to LDA_D, that finds D to be TLR-reachable. TRR terminates with a high probability, if D has not moved away from the location known at the source more than the scope of an average TLR region (200 m). Therefore, in order to terminate TRR, the source should have information about destination location that is at most 10 seconds old.

Provided that the end-to-end delay of the packet is small, terminode routing requires that the destination update interval is 10 seconds.

4.11.2 A Location Service suitable for a Terminode Network

Location service in a terminode network should distribute location information dynamically inside the network, with a minimum of communication involved in upgrading and in the retrieval of the location information. The main functions of the location service are: (1) maintaining the location information, and (2) distributing the location information inside the network. The requirement for location service is that all control messages (those use for building the location service location updates and queries) for the location database are routed using terminode rout-

ing itself. In this section we give an overview of the VHR location service [68] that is designed to scale in a larger scale mobile ad hoc network and thus is suitable for the location discovery service in a terminode network. The integration of terminode routing with the VHR location service is left for future work.

VHR proposes that each node advertises its current position (LDA) to a geographical region called Virtual Home Region (*VHR*). The VHR has a fixed centre C_{VHR} and a variable radius that adapts to the density of the area containing the VHR, in order to maintain an approximately constant number of nodes inside the VHR. The fixed centre is computed using some predefined, publicly known hash function H. H is a static mapping between the space of EUIs $\{EUI_i\}$ and the geographical space of a node network such that: $H(EUI_A) = C_{VHR}$ for each $A \in \{EUI_i\}$. Node A updates its location by sending position advertisements towards its VHR. All nodes in A's VHR store the mapping between A's EUI and its LDA. They act as location servers for A during the time they belong to A's VHR; once they leave it, they loose this role. When a node B wants to retrieve the location of A (it must know A's EUI), it sends a query towards A's VHR (which can be computed by B since the hash function is publicly known). Once this request is received, the location information of A is sent back by the nodes of the VHR of A. A drawback of VHR, is that terminode B that need to know a location of a close terminode A, may contact a potentially distant VHR of A. One possible solution to this problem is to organize the VHR-based mobility management in a hierarchical way in order to scale better in a large network. In this solution, every terminode would have several geographically distributed VHRs that contain its location. For retrieving location information of another terminode, a terminode would contact the corresponding VHR that is closest to it. There are several possible location updating schemes that can be applied in a terminode network. The following schemes are proposed for personal communications networks(PCN)[81] and can be used in a terminodes network. In the *timer-based* location update scheme, each node periodically sends a location update to its VHR. Having in mind the requirements of terminode routing on the location management, the period for sending location updates should be 10 seconds. In the *distance-based* update scheme, each terminode tracks the distance it has moved since its last update and sends its location update whenever the distance exceeds a certain threshold. In a terminode network threshold should be equal to a nominal transmission range. In the *predictive distance-based* scheme, the terminode reports to its VHR both its location and velocity. Based on this information, and a mobility pattern, the location of the terminode can be predicted. The terminode checks its location periodically and sends a location update whenever the distance between the predicted location and its exact location exceeds the given threshold. Again, for a terminode network threshold should be equal to a nominal transmission range (250 meters).

4.12 Multipath Terminode Routing

In this section we advocate that sources normally attempt to acquire and maintain several paths to destinations that they communicate with. As said before, these paths may either be greedy or anchored paths. Then, sources should distribute their flows into multiple paths. We describe the architecture of multipath terminode routing. Implementation and analysis of multipath terminode routing is outside the scope of this thesis.

4.12.1 The need for multipath routing

The reasons for using multipath routing are the following:

- Multipath routing is a way to cope with uncertainty in a terminode network; the paths that a source has acquired can deteriorate due to mobility and packets can be lost. The use of multiple disjoint paths provide better load balancing and path failure protection.

- Routing in a highly dynamic network without a fixed infrastructure should be different from the traditional routing. Traditionally the shortest path routing is applied in the global Internet. Internet has a characteristic hierarchical topology; at the higher level is the backbone. Most of the packets in the network are routed via the backbone links. When the traffic demands are increased, the backbone links may become saturated. Because the backbone links are fixed, network management can easily respond to expansion of the network. In this case it should be sufficient to uniformly upgrade the backbone links in order to manage increasing traffic demands.

  Within a mobile ad hoc network, network topology is highly dynamic and unpredictable (there is not a fixed backbone). On one hand, it is hard to discover and maintain shortest paths between nodes in the network. On the other hand, even if the shortest path is found it may be composed of links that may be congested. It is not possible to increase the capacity of links as a response to increasing traffic demands. Therefore, traditional shortest path routing cannot be applied in mobile ad hoc networks.

  As we have already explained, terminode routing uses nodes' positions in order to find paths in the network. An anchored path is not defined with a list of nodes, but an anchored path, roughly speaking, defines regions that the packet should pass in order to reach the destination. We described in Section 4.9.2 one possible strategy for finding anchored paths when the network map is given: terminodes use the network map to determine the shortest

anchored path between the source and destination town. This strategy may have the following problem: if many anchored paths pass through the same region, it may happen that that some regions become congested. This problem was also reported in our simulation results in the next chapter.

- Because the traditional shortest path routing is not applicable is a mobile ad hoc network, routing in an ad hoc network should be dynamic: it should cope with uncertainties in the network in a dynamic way. In Appendix A, we present the theoretical solution to the dynamic routing problem. The dynamic routing in the network has as the objective to maximize the global network utility. In Appendix A we show that multipath forwarding ensures that the utility of the network approaches optimum; the optimal path is a multipath.

4.12.2 Path maintenance and multipath routing in a terminode network

In this section we give directions on path maintenance and multipath routing in a terminode network.

In Appendix A for the purpose of the work on the optimal routing, a terminode network is presented, in a macroscopic way, as being composed of a number of blocks populated by terminodes. Each block is characterized with its capacity. The global utility is given as the sum over the user utilities minus the cost imposed by the network [42], under the constraints imposed by the transit capabilities of the blocks that constitute a terminode network. We look at the routing as an optimization problem, and we want to jointly optimize flow control and routing, such that the global utility function is maximized. A route in a network is defined with a list of blocks to be passed from the block in which the source resides to the destination block. Theoretical results in Appendix A indicate that the optimum of network utility can be achieved under the assumption that all nodes in the network always had a complete view of the network topology, of capacities and of shadow prices of all blocks that build the network. Then the solution to the optimal routing problem gives that sources should be rate adaptive, and they should distribute their flows over the routes that have the smallest *route shadow price*. The route shadow price can be interpreted as a congestion feedback information from the network to the source when the source sends its flow over the route. Source can then use multiple paths to send its rate. Then, the source should split this rate *arbitrarily* among paths that have the same route shadow price.

The assumptions that are used for solving the optimal routing problem in a network modeled with blocks are not satisfied in a real terminode network. The reason is that terminodes do not

always have a complete view of the network topology, of capacities and of shadow prices of all blocks that build the network. We used the theoretical result to suggest how the sources in a terminodes network should behave in order to tend to the global optimal. However, the distributed algorithm that every terminode locally runs in order to solve a global optimization problem is a matter for future work.

The source acquires several anchored paths to its destination. The number of anchored paths may depend on available resources and the importance of the destination. For example, the terminode may invest more to discover more paths to the destination with which it communicates often, or with which it has a long communication. One of the paths that the source can use is the path without anchors (greedy path to the destination). The method presented in Section 4.10 can be used by sources to evaluate whether the greedy path to the destination is acceptable. In addition, the source can acquire several anchor paths to the destination. In the case when the network topology is known to all nodes (networks maps are available or can be built) then several paths that connect the source to the destination can be obtained from maps (Section 4.9.2). In the case when friends are used for path discovery, the source can start in parallel the FAPDP protocol with several friends in order to acquire several anchored paths (Section 4.9.1).

All anchored paths are maintained. Path maintenance consists of four main functions: independent path selection, path simplification, path monitoring and deletion, and congestion control.

independent path selection A terminode analyzes all acquired paths to a destination. Then its selects a set of independent paths. They are paths that are as diverse as possible in the geographical points (anchors) that they consist of. Diversity of paths is essential for taking advantage of multipath routing [63].

path simplification One method consists in approximating an existing path with a path with fewer anchors. Such an approximation yields a candidate path, which may be better or worse than the old one.

path monitoring and deletion A terminode constantly monitors existing paths in order to collect necessary information to give the value to the path. The value of the path is given in terms of congestion feedback information such as packet loss and delay. Other factors like robustness, utilization, stability and security are also relevant to the value of a path. This allows a terminode to improve paths, and delete mal-functioning paths or obsolete paths (e.g, the path that corresponds to two terminodes that do not communicate any more).

congestion control The value of the path given in terms of congestion feedback information is used for the source to decide how to split the traffic among several paths that exist to

the destination. Congestion feedback information that corresponds to one path can be interpreted as a shadow price of the path. With a source having in hand a set of paths to a destination, the source need to control the use of these paths. First, a source selects a set of paths that give least least congestion feedback information (minimum shadow price). Second, if there are several paths with a minimal shadow price, a source distributes its traffic among multiple routes (e.g., the source sends the equal amount of data on each of these routes). For example, a simple per-packet allocation scheme can be used to distribute the data among available routes. If all terminodes behave in this way, our belief is that approaching the optimum of the utility of the network,will be achieved. Implementation and evaluation of multipath terminode routing is the subject of future work.

Note We assume that multipath routing is acceptable for the transport protocol. However, with the current TCP this is not acceptable because there are problems with managing a large number of timers due to many paths. We envision either to bring enhancements to the current TCP, or to use multiple description coding techniques [48]. In this latter case, the source sends out a stream of encoded packets that constitute an encoding of the source data. These packets are distributed over multiple anchored paths that a source maintains to the destination. We can use erasure coding at the source that takes a block of source data consisting of k packets and produces a sufficient amount of encoding packets (n). Erasure codes have a property that allows for a decoder at the client side to reconstruct the original source data whenever it receives any k of the n transmitted packets. Data recovery is performed at the destination in the presence of a limited number of packet losses along multiple paths. After receiving a sufficient k number of data packets to do data decoding, the destination sends a feedback to the source. If the feedback comes before the time it takes a source to send all n packets, a source can stop sending more data packets of the block of n packets.

4.13 Conclusion

In this chapter we focused on the problem of routing in a wide-area mobile ad hoc network called a Terminode Network. Routing in this kind of network is designed with the following objectives. First, it should scale well in terms of the number of nodes and geographic coverage. Second, routing should have scalable mechanisms that cope with the dynamicity in the network due to mobility. Third, routing should not be based on complex algorithms or protocols that would require a high routing load. Our routing scheme is a combination of two protocols called

Terminode Local Routing (TLR) and Terminode Remote Routing (TRR). TRR is activated when the destination D is remote and it uses the location of the destination obtained either via location management or by location tracking. TLR acts when the packet gets close to the destination and uses routing tables built with hello messages. The use of TRR results in a scalable solution that reduces dependence on the intermediate systems, while TLR allows to increase the probability of reaching the destination, even when it has moved considerably from the location that was known at the source. TRR has two packet forwarding methods: Geodesic Packet Forwarding (GPF) and Anchored Geodesic Packet Forwarding (AGPF). GPF is basically a greedy method that forwards the packet closer to the destination location until the destination is reached. AGPF forwards packets along anchored paths. An anchor path is a list of fixed geographical points that define regions where packets should be forwarded in order to avoid holes in terminodes distribution. We propose two algorithms for anchored path discovery: the first one is called Friend Assisted Path Discovery (FAPD), and the second is called Geographic Maps-based Path Discovery (GMPD). Finally, we outline how multipath terminode routing is applied in order to achieve better load balancing and path failure protection.

Chapter 5

Performance Evaluation of Terminode Routing

5.1 Introduction

In this section we validate the operation of terminode routing by using simulations. Our objective is to test how terminode routing performs in numerous simulation environments: in different sized mobile ad hoc networks, under different node mobility or under different load in the network.

This chapter is organized as follows. Section 5.2 describes terminode routing implementation. Section 5.3 gives terminode routing performance evaluation in a small mobile ad hoc network where nodes are uniformly distributed. In Section 5.4 we use simulations to show that the terminode routing methods, TLR and RLF, help to alleviate problems due to location inaccuracy. Section 5.5 illustrates simulation results in a larger mobile ad hoc network, where terminodes are not uniformly distributed.

We implemented and simulated the terminode routing protocol in GloMoSim[74]. GloMoSim is a scalable simulation environment for wireless network systems. It is based on the parallel, discrete-event simulation language PARSEC[4]. GloMoSim models the OSI seven-layer network architecture and includes models of different MAC protocols, IP routing, UDP and TCP. The simulator also models node mobility, therefore providing simulations of mobile ad hoc networks. We have selected GloMoSim as the simulator due to its inclusion of various models, its scalability and ease of operation. NS[1] is another simulator that is used for simulating mobile ad hoc networks. We selected GloMoSim instead of NS because GloMoSim requires considerably less running time than NS. We want to perform simulations of large mobile ad hoc

networks, and we found GloMoSim to scale much better than NS.

5.2 Terminode Routing Protocol Implementation

In our implementation of terminode routing, we made several design choices:

5.2.1 HELLO Messages

As explained in Section 4.4, terminodes periodically broadcast HELLO messages. The information contained in HELLO messages is used for terminodes to build their local routing tables. Every node maintains a HELLO timer that is set to one second. When a HELLO message is sent, a HELLO timer is reset. Each entry in the routing table expires after two seconds, if it is not updated. We verified by simulations that the chosen interval for sending HELLO messages ensures accurate neighbours' list for a large scale of mobility degrees. In particular, we examined nodes' velocities in the range 0 to 20 meters per second.

In order to reduce the routing overhead, caused by sending HELLO messages, we make promiscuous use of the network interface. Then, by disabling MAC address filtering, nodes receive all packets from all nodes in their radio range. DSR [12] and GPSR [39] also run their interfaces in the promiscuous mode. In the case of DSR, nodes learn promiscuously about source routes in the network. In the case of GPSR, all packets carry their local sender's position. Then a node that sends a packet defers sending a HELLO message (beacon). The effect of this reduces of the rate at which beacon packets must be sent, and keeps positions in neighbour lists maximally current in regions under traffic load.

In our implementation, promiscuous running of network interfaces is used in a similar way as in GPSR. Nodes that have data packets to send should defer sending HELLO messages, because data packets piggyback the HELLO message information. However, in our implementation, not all data packets piggyback the HELLO message information. Because a node does not considerably change its location and its neighbours list, between two data packets, it is not meaningful to increase the size of all data packets with the HELLO message control information. We do as follows. A HELLO timer is set to 0.5 seconds. When a HELLO timer expires, if a node has a data packet to send within 0.5 seconds from the time when a HELLO timer has expired, the data packet piggybacks the HELLO message information, and a HELLO message is not sent, otherwise, a HELLO message is sent. In both cases, the node resets its HELLO timer. To avoid synchronization of HELLO messages, a terminode jitters each HELLO message transmission by a random interval between 0 and 1 second.

5.2.2 Support for MAC-layer Failure Feedback

802.11 MAC layer notifies the network layer when a unicast packet exceeds the maximum number of retransmissions and the acknowledgement has not arrived. This means that the intended neighbour has left the sender's transmission range. As a consequence, the entry that corresponds to that neighbour is invalid and is thus removed from a sender's neighbour table. In our implementation such a packet is sent back to the routing layer where a new neighbour (to send a packet to) is chosen.

5.2.3 Location Information

In our implementation every terminode knows accurately its current location all the time. Terminode routing uses the destination location for packet forwarding. Location management is used at sources to learn the destination location.

Location management in a small mobile ad hoc network In the simulations of small ad hoc networks, we include the simple location management that works as follows. Sources learn about the destination location on demand. When the source has some data to send to the destination whose location is not known, a flooding based approach is used for destination location discovery. The method is similar to DSR source route discovery [12]. Once the communication between source and destination is started, location tracking is used for updating destination location.

location discovery

When source S has data to send to destination D that is not reachable by TLR, S needs to find the location of D. S buffers all data packets until it learns D's location. To do so, S broadcasts a *location request* control packet to all its neighbours. Inside the packet, S stamps its own location. Node X, which receives a location request packet and is not the destination, broadcasts the request to its neighbours. In order to avoid a redundant transmission of the request, X should broadcast a particular location request packet only once. A source of a location request control packet stamps the packet with a sequence number. Intermediate nodes keep a cache of already seen location request packets. Entries in this cache are kept for 30 seconds. An already seen location request packet is discarded. On receiving the location request, destination D responds to S with the *location reply* control packet. The location reply carries D's location. D sends the location reply back to S by GPF, using the S's location (that D learnt from the location request). Upon reception of the location reply, S stores D's location in its location cache and sends buffered data packets by using GPF (see Section 4.6). But, if S does not receive a location

reply from the destination after 2 seconds, S initiates again the flooding of the location request control packet with the new sequence number.

location tracking

Once the two nodes begin communication, location tracking is used: data packets periodically piggyback the local sending node's position. We performed simulations with only CBR traffic sources. Hence, destinations nodes periodically send to corresponding sources a *location update* control packet with their current location. The period for sending location updates is set to 5 seconds. The destination stamps each location update packet with a sequence number. The source updates the destination location upon reception of a location update packet with a higher sequence number. If the destination location entry is not refreshed for more than 10 seconds, the entry is removed, and the source re-initiates learning of the destination location through flooding.

We note here that the described location management scheme is not applicable within a larger mobile ad hoc network because it includes flooding of the network.

Idealized location management in a large mobile ad hoc network In the simulations of larger mobile ad hoc networks we do not include a distributed location database for annotating packets with destinations' positions. We assume an idealized location database where all nodes can know all other nodes' positions at all times with no control overhead. However, a source does not stamp data packets with the true location of the destination at all times. We examine the terminode routing performance when there are inaccuracies in location information; We assume that the source cannot know an exact destination location all the time. In our simulations, the source learns a destination location and uses this information for the time that we call *location information lifetime*. After this time, the source again acquires an exact destination location and uses it for another location information lifetime interval. Location information lifetime is a parameter can be set at the beginning of the simulation.

Simulation Environment

The IEEE 802.11 Medium Access Control(MAC) protocol is used; it implements the Distributed Coordination Function (DCF)[15]. In all simulations, radio range is the same for every terminode, and is equal to 250 meters. The channel capacity is 2Mbits/sec. This corresponds to characteristics of Lucent Technologies WaveLAN [13]

The propagation model, included in the GloMoSim simulation package, is the two-ray model. It uses free space path loss for near sight and plane earth path loss for far sight.

5.3 Evaluation in a small network with uniform node distribution

The goal of this section is to compare by means of simulations the performance of terminode routing versus two other routing protocols, AODV and LAR1 (LAR scheme 1), in a small ad hoc network. In our simulations we use the rectangular unobstructed simulation area of the size 2200 m X 600 m with 100 nodes. Nodes in the network are uniformly distributed; nodes are free to move in the whole simulation area according to the mobility model presented below.

AODV and LAR1 are chosen because they perform very well for a small ad hoc network, and they are based on different routing strategies. As we mentioned in Chapter 3, AODV does not use geographic positions. The control part of LAR1 uses geographic positions, while packet forwarding in LAR1 uses source routes as in DSR. Simulations of AODV and LAR1 are performed using the implementations that are included with the GloMoSim simulation package. The relevant AODV and LAR1 simulation parameters are given in Appendix C.

Because the simulation area is small and unobstructed, terminode routing uses only the GPF packet forwarding method (see Section 4.6). The simulated network is densely populated. The density of the network (75 nodes per square kilometer) ensures that GPF forwards most of the packets in a greedy mode. Nodes positions that are used in GPF are obtained using the location management method that we described in Section 5.2.3.

Mobility Model The mobility model is the "random waypoint" mobility model[12]. In this model a node chooses one random destination in the simulation area. Then it moves to that destination at a random speed (uniformly distributed between 0-20 m/sec). Upon reaching its destination, the node pauses for the *pause time*, selects another random destination inside the simulation area, and proceeds as previously described. In our simulations we vary the pause time, which affects the relative speed of mobile nodes.

Communication Model Traffic sources are CBR (constant bit-rate). The source-destination pairs are randomly spread over the network. All data packets are 64 bytes long. We performed simulations with 20 and 40 source-destination pairs in order to change the offered load to the network. The packet rate is fixed at 4 packets/sec for 20 sources, and 2 packets/sec for 40 sources. In both cases, the offered load to the network, in number of packets, is approximately the same. However, in the case of 40 sources, there are two times more source-destination pairs than for 20 sources. The flows are low-bitrate, and the network is not congested, because these

simulations are meant to measure routing protocols behaviour, not the limitation of the IEEE 802.11 MAC for data packet capacity.

Simulations are run for 900 simulated seconds. Each data point represents an average of six runs with identical traffic models, but different randomly generated mobility scenarios. For all simulated protocols we use identical mobility and traffic scenarios.

We looked at three performance metrics that are used also in [67]:

Packet delivery fraction, the ratio of the data packets delivered to the destinations to data packets generated by the CBR sources,

Average end-to-end delay of data packets, which includes all possible delays caused by queuing, retransmissions at the MAC, propagation and transfer time. In the cases of AODV and LAR1, this also includes delays caused by buffering during route discovery. In the case of terminode routing, this includes delays caused by packets buffering during the destination location discovery.

Normalized routing load, the number of transmitted routing (control) packets per data packets delivered at destinations. In the case of AODV and LAR1, control packets are route request, reply and error packets. Route request packets are generated by sources and flooded in the whole or a part of the network, route reply and error packets are generated by destinations and forwarded to packet sources. Terminode routing generates four types of routing packets: HELLO messages that are generated periodically but not forwarded more than one hop; location request packets, generated by sources when the destination address is needed, and flooded to the network; location reply packets are generated by destinations and forwarded to sources upon reception of the location request; and location reply packets that are periodically generated by destinations and forwarded to packet sources. Each hop-wise transmission of a routing packet is counted as one transmission.

Simulation Results

Our simulation results show the following.

Terminode routing outperforms AODV and LAR1 in all experiments. In the set of experiments with 20 sources, all three protocols are successful in delivering more than 80% of data packets. Terminode routing and LAR1 deliver more data packets than AODV. Even at lower pause times (higher mobility) terminode routing succeeds in delivering more than 95% of data packets.

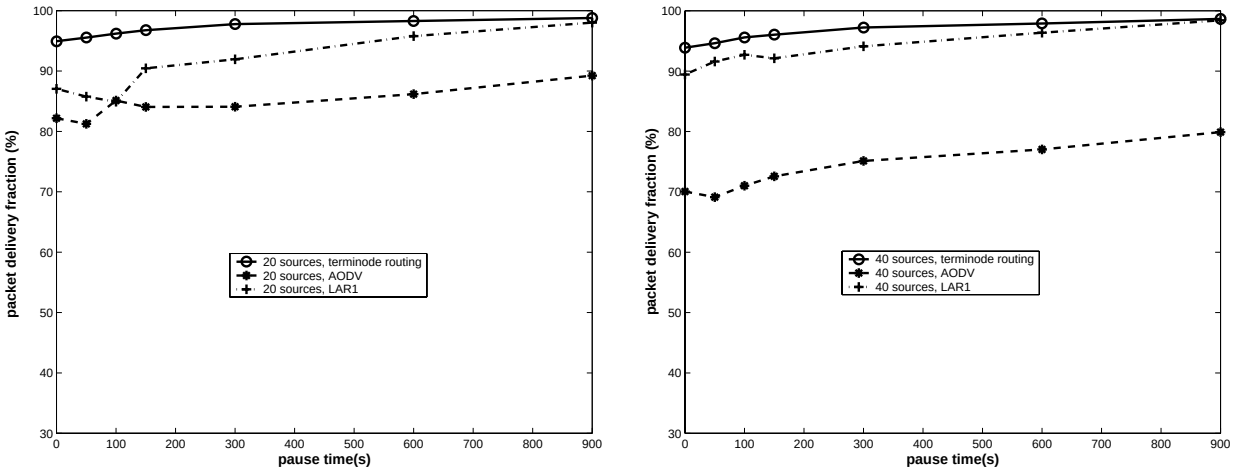
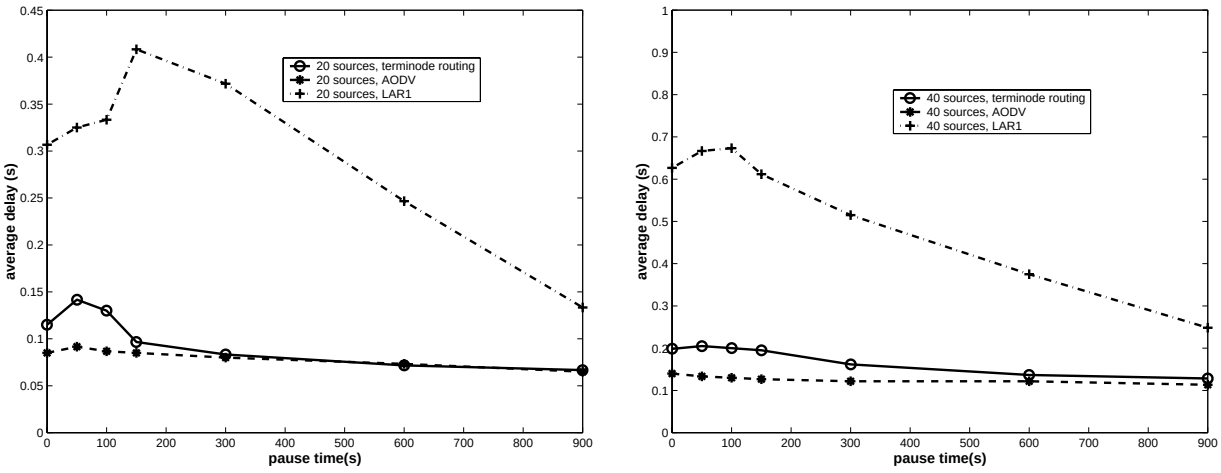


Figure 5.1: (a) Packet delivery fraction for the 100 node model and 20 sources , (b) Packet delivery fraction for the 100 node model and 40 sources

In the set of experiments with 40 sources, terminode routing and LAR1 again performed very well, however AODV reduced its packet delivery rate.

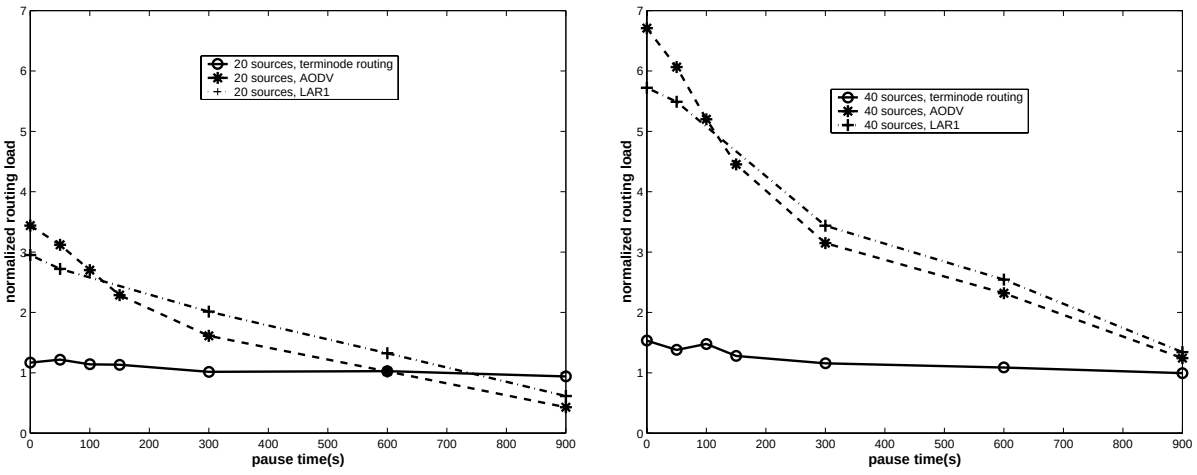
The delay experienced by LAR1 is higher than the one of AODV and of terminode routing. When terminode routing is used, the delay due to buffering during the destination location discovery is critical in the initial phase when the source learns about the destination location. If the location management works well, and the source regularly receives destination location updates, packets are not buffered at the source waiting for the destination location. In the case of AODV and LAR1, delays caused by packet buffering during the route discovery are present every time a source has to (re)discover a route to the destination.

In all experiments, terminode routing has the smallest normalized load. Moreover, terminode routing has a stable normalized routing load for different pause intervals. There are two reasons for this property: First, every node proactively generates HELLO messages every second, unless a data packet is sent, and these messages are received, but not forwarded by neighbours. The overhead due to HELLO messages is independent of the mobility rate of nodes and the number of traffic flows. Second, routing overhead due to location management does not change very much with the increase of mobility. Location update control packets are generated proactively by destinations and independently of mobility. We verified by simulations that in most cases location request packets flood the network only at the beginning when the communication between source



(a) (b)

Figure 5.2: (a) Average data packet delay for the 100 node model and 20 sources , (b) Average data packet delay for the 100 node model and 40 sources



(a) (b)

Figure 5.3: (a) Normalized routing loads for the 100 node model and 20 sources , (b)Normalized routing loads for the 100 node model and 40 sources

and destination starts. After that, mobility tracking (with sending of location reply packets) ensures that the source receives periodic updates of the destination location, without need to often flood the network. As the number of sources increases, terminode routing generates more routing overhead due to location management control packets, but we observe a slight increase of routing overhead with the increase of the number of data sources.

LAR1 builds source routes while AODV relies on routing tables at each node in order to reach the destination. In both protocols, when a single link in the built route is broken, a new route should be built. Both AODV and LAR1 include flooding in order to build new or repair broken routes. LAR1 floods to the expected zone, while AODV does an expanded ring search type of flooding. For small values of pause time (higher mobility) more routes are broken and in order to repair them, AODV and LAR1 generate more routing overhead for higher mobility.

Both AODV and LAR1 maintain routes to destinations. We verified, by simulations, that maintaining routes with many hops in mobile ad hoc networks is a difficult challenge. Terminode routing does not build the route to the destination; routing decisions are made locally at each terminode. We showed by simulations that this strategy is better than the strategy of building routes - provided that node density is high and sources can acquire accurate destinations location.

5.4 Evaluation of usefulness of Terminode Local Routing (TLR) and of Restricted Local Flooding (RLF) in the case when accuracy of location information is low

Example of usefulness of TLR

Figure 5.4 illustrates with one example the benefit experienced if TLR is used when the packet gets close to the destination. Assume that the packet, whose destination is D, arrives at terminode A. A finds the destination D within its local routing table and thus D can be reached with TLR even if D moved considerable distance from the used location (LDA_D). With TLR, A sends the packet to F, and F forwards the packet to the destination.

If TLR is not used, then packet forwarding is done based only on locations. A sends the packet to B, because B is closest to LDA_D. Similarly, B sends the packet to X. X finds that the condition to expedite the termination of TRR is met (LDA_D is within the transmission range of X). In this way, eventually, the packet may arrive at the destination. Because of the many hops it has to pass, it is more probable that the packet will be dropped. With TLR, even if the destination

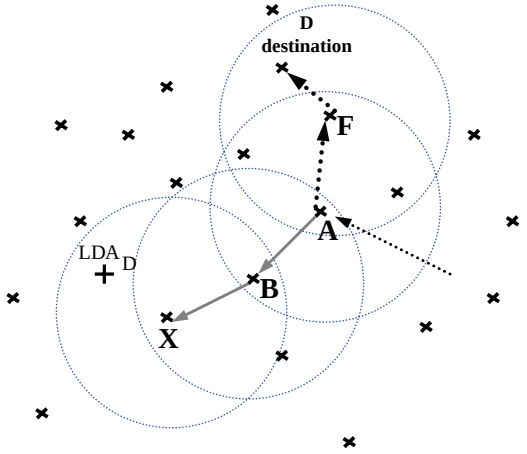


Figure 5.4: Illustration of TLR utility; dash lines correspond to TLR packet forwarding

has moved a considerably from its initial position, the packet can arrive at the destination within less number of hops.

To evaluate the benefit of TLR by simulations, we analyze the performance of terminode routing when TLR is used and when it is not. In the first simulation case study, TLR is not used. There, nodes do not maintain their TLR-reachable areas: GPF is used to forward packets from the source to the destination.

In the second simulation study case, TLR is used. Every terminode maintains a list of its TLR-reachable destinations. Similar to the first case, GPF is used to forward packets from the source to its non-TLR-reachable destination. But, when some intermediate node finds that the destination is TLR-reachable, it uses TLR to send the packet to the destination. Note that TLR is used in a two-hop neighborhood and does not need additional routing overhead compared to the case when TLR is not used. The only additional requirement when TLR is used, is that all terminodes keep in their routing tables information not only about immediate neighbours, but also about their two-hop neighbours.

In our simulations we use a large network of 600 terminodes, with the uniform node distribution. The simulation area is a square of the size 2900m X 2900m. The simulated network is quite dense; in this case we verified that GPF mostly forwards packets in the greedy mode. We simulated 20 CBR traffic flows. Each CBR flow sends two packets per second. Only 64-byte packets are used. Terminodes move according to to the “random waypoint” mobility model. In our simulations, a speed is uniformly distributed between 0-20m/s and the pause time is 10s.

In these simulations for a larger mobile ad hoc networks we do not include a distributed location database. However, we use the *location information lifetime* parameter to denote the

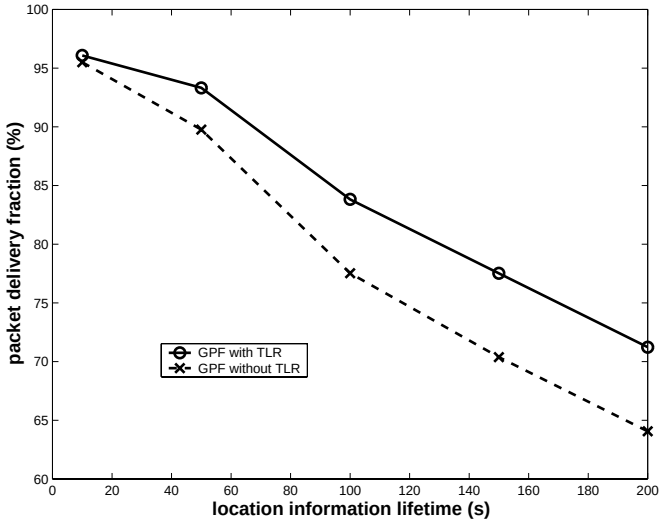


Figure 5.5: Figure shows that using TLR results in higher packet delivery fraction than in the case when only position based routing is used

time interval as which the source learns the exact destination location.

The two study cases (GFP with TLR, and GPF without TLR) are evaluated for different values of location information lifetime parameter. We simulated six different randomly generated motion patterns. Figure 5.5 presents an average of packet delivery fraction for five simulation runs. This figure shows that for smaller location information lifetimes (less than 20 seconds), the packet delivery fraction is similar with TLR and without TLR. However, for higher location information lifetimes (lower precision of location information) routing with TLR gives better delivery fraction than without TLR. Therefore, we conclude that when using TLR, routing is more robust in the case of positional errors and inconsistent location information. With the size of the TLR-area equal to two hops, routing continues to successfully deliver packets to destinations even if the location management is not able to provide the locations updates more frequently than one minute.

Example of usefulness of RLF

In all the simulations presented so far, when the packet arrives at some node where conditions for expedited termination of TRR are met (because the packet arrives close to the destination's location stamped in the packet, but the destination is not TLR-reachable), the lifetime of the packets is limited to $term\_trr$ hops ($term\_trr$ is equal to 3 hops). Then if the packet does not arrive at the destination, the packet is dropped.

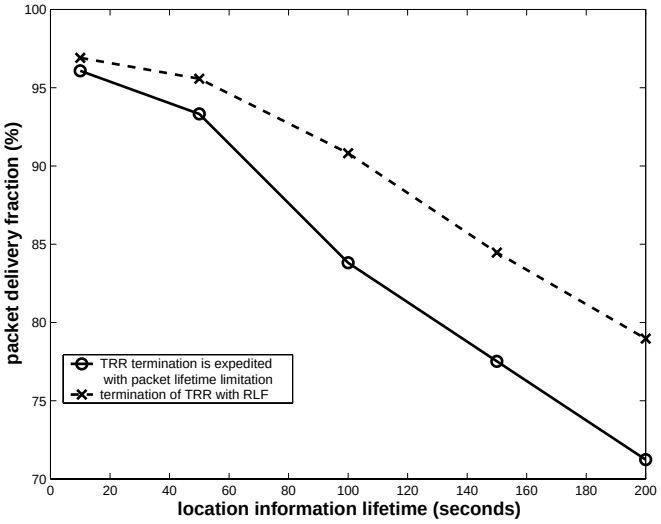


Figure 5.6: When Restricted Local Flooding is performed, fraction of received packets is higher than without RLF (when TRR is terminated by the packet lifetime limitation).

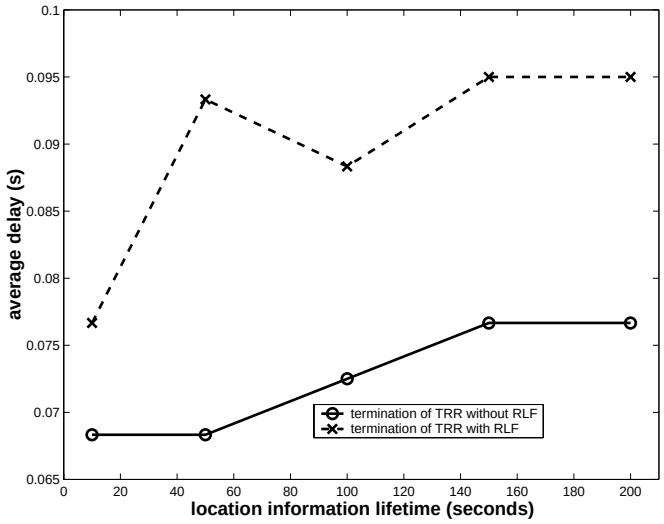


Figure 5.7: When Restricted Local Flooding (RLF) is performed, the average delay is higher than without RLF (when TRR is terminated by the packet lifetime limitation)

Now, we evaluate by simulations the Restricted Local Flooding (RLF) method, which is the second method that is used to terminate TRR (see Section 4.8). We use the same simulation settings as in previous simulations that evaluate TLR.

Recall from Section 4.8, that node X that initiates RLF, sends six copies of the packet towards six different geographic points around X. In this way, we increase the expected region where destination can be found. Therefore, we increase the probability that some of the flooded packets will arrive at the destination. Figures 5.6 and 5.7 compare packet delivery fraction and the average packet delay of two possible ways of TRR termination: in the first case termination of TRR is done by limitation of lifetime of a packet, and the second case corresponds when RLF is used. Figures 5.6 illustrate the improvements in the fraction of delivered packets when RLF is used compared to the case when RLF is not used. We see that RLF is especially beneficial for larger values of location information lifetime (e.i., the destination location is less accurate because the source gets the destination location updates less frequently).

However, the use of RLF is not without cost. Figure 5.7 illustrates increased average packet delay when RLF is used. The reason are longer paths of a number of packets to reach the destination. These packets are otherwise dropped in the case when RLF is not used.

5.5 Evaluation of Anchored Geodesic Packet Forwarding (AGPF) in a large ad hoc network with non-uniform node distribution

AGPF (see Section 4.7) is evaluated within a relatively large simulation area where nodes are not uniformly distributed. This means that there are regions within a simulation area where nodes cannot move to, and thus there are holes in terminodes distribution.

We analyze AGPF by comparing a performance of terminode routing when the combination of AGPF and GPF is used and when only GPF is used (see Section 4.6).

We do not provide results for other protocols such as DSR, LAR and AODV, because they are designed for relatively small scale networks and the comparison would not be fair. Recall from Chapters 3 and 4 that GPSR [39] and GPF are similar. Note that the only difference between the two protocols is the following: GPSR uses the destination location for making packet forwarding decisions for the whole way until the packet arrives at the destination; with GPF, an intermediate node switches to TLR if the destination is TLR-reachable. In the case when GPF cannot be terminated because the destination has moved from its reference position and no terminode finds

the destination to be TLR-reachable, GPF termination is done by limitation of lifetime of a packet. Therefore, by the comparison of AGPF against GPF we also evaluate AGPF against GPSR.

Our simulations do not include a distributed location database for annotating packets with destinations' position. We assume an idealized location database where all nodes can know all other nodes' positions at all times with no control overhead. However, a source does not stamps data packets with a true location of the destination at all times. The *location information lifetime* parameter is used in our simulations to denote the time interval at which the source can learn the exact destination location. An idealized location database system has no network or computational cost. Hence, our simulation results may be better than one might expect with a real location service.

Before presenting the simulations and the results, we introduce the new mobility model called "restricted random waypoint" that is used in evaluation of AGPF.

5.5.1 Restricted Random Waypoint Mobility Model

In the most recent papers about mobile ad hoc network simulations, nodes in the simulation move according to the "random waypoint" model as described in Section 5.3.

We find this model unrealistic for a large mobile ad hoc network, such is a terminode network. In this network, terminodes are small personal devices that are distributed geographically within a very large area. It is less probable that, for each movement, a terminode selects a random destination within a very large geographic area. On the contrary, the random destination is selected within a small area for a number of movements, and then a movement is made over a long distance. This better represents the fact that most people move for a certain period within one area, and then they move away to another distant area. We have implemented a new mobility model that we call "restricted random waypoint". This model is closer to a real-life situation for a wide-area mobile ad hoc network than the random waypoint model.

For the restricted random waypoint mobility model, we introduce a topology based on towns and highways. Towns are areas that are connected with highways. Inside town areas, terminodes move with the random waypoint mobility model. After a certain number of movements in the same town, a terminode moves to another town. Terminodes that move between the town areas, simulate highways between towns. The model of the simulated area that consists of four towns is presented in Figure 5.8. In our simulations we define, inside the configuration file, the pairs of towns that are connected by highways. For example in Figure 5.8 those pairs are (town 0, town 1), (town 0, town 2), and (town 1, town 3). This information is used by terminodes when they

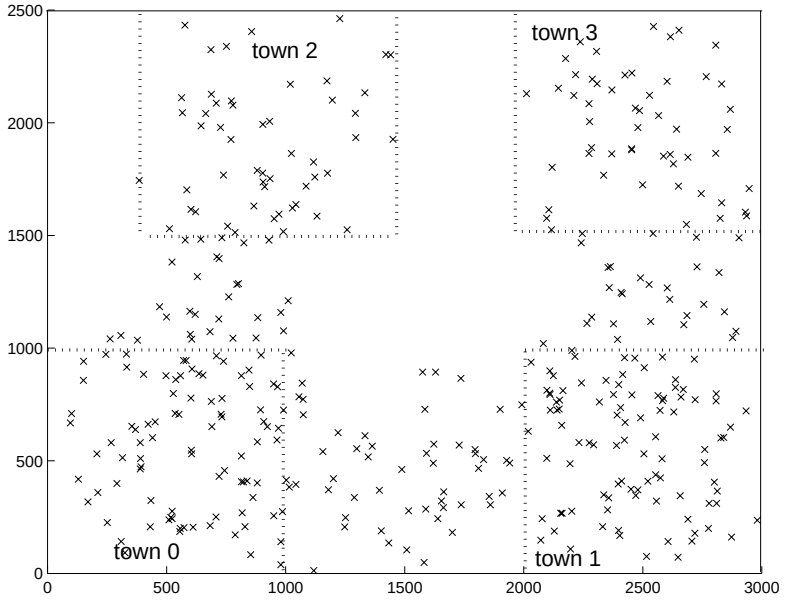


Figure 5.8: Model of the simulation area with four towns

move from one town to another.

We distinguish two types of restricted mobility that represent the “ordinary terminode” and the “commuter terminode”.

At the beginning of the simulation, terminodes are placed at randomly chosen locations inside one of four towns. An “ordinary terminode” begins the simulation by selecting at random one destination inside the town where it is placed. Then it moves to that destination at the speed distributed uniformly between 0 and some maximum speed. Upon reaching that destination, the ordinary terminode pauses for the *pause time*, selects another destination within the same town, and proceeds as previously described. Thus, the ordinary terminode’s movement inside a town is the random waypoint mobility model. It repeats such movements for a number of times set by the *stay_in_town* parameter. Then a terminode selects at random a destination within a new town and moves there (the new town is randomly chosen from a list of towns that are connected with the current town by a highway). Once it reaches the new location, a terminode applies inside the new town the random waypoint mobility model for another *stay_in_town* time.

There are also a number of terminodes that frequently commute from one town to another. Those terminodes are called “commuters” and they ensure the connectivity between towns. The commuter’s movement model is the restricted random waypoint where *stay_in_town* parameter is equal to one. A commuter selects a random destination within one town area and moves to

that destination with a speed distributed uniformly between some minimum speed and some maximum speed. Once this destination is reached, a commuter pauses for a pause time that is smaller than an ordinary terminode's pause time. Then it selects at random another town (so that is connected with the current town) and the random destination inside the chosen town, and moves to this destination. It pauses in the new town for a small interval of time and then moves again to another town.

An example of a network that comes out when terminodes are moving according to the restricted random waypoint mobility model is presented in Figure 5.8. In such a network, not all town areas are connected with a highway (e.g, town 2 and town 3).

5.5.2 Scenario Characteristics

In order to assess the relevance of AGPF, we use simulations to evaluate packet forwarding in two cases: the first case corresponds to when the combination of GPF and AGPF is used. The second case is when only GPF is used; in this case the source stamps the destination location in the packet header, which is the only location information used by intermediate nodes to forward packets to the destination.

When AGPF is used, the GMPD method is used for anchored paths discovery (see Section 4.9.2). In the simulation model based on "towns and highways", we assume that a high level geographic view of the network is available at every terminode. This means that each terminode has a knowledge of a map of towns. A map defines town areas and the existence of highways between towns. Thus, for example, the map of towns presented in Figure 5.8 defines those towns that are directly connected by highways as well as towns' areas. In our simulations, a town area is a square around the town center with the given width around the town center.

When source S has some data to send to destination D, S first determines the "destination town" (DT). This is the town in which area D's location falls. If D is not inside any town area, then the destination town is the town whose center is closest to D's location. Similarly, S determines the "source town" (ST), the town where S is situated, or the closest town to S if S is on the highway.

Once S determines DT and ST, S contacts the map of towns to check if DT and ST are the same, or if they are directly connected with a highway. If so, then geodesic packet forwarding (GPF) towards D has a good chance of working. Then, S does not add an anchored path to the packet and S sends a packet using GPF.

Otherwise, if there is no highway from ST to DT, S finds out from the map those town areas that a packet has to pass in order to reach D. Then S adds to the packet the anchor path. This

anchored path is given by a list of the centers of towns that the packet has to pass. Then S starts AGPF in order to deliver the packet.

For example, if S is in the area of town 2 and D is the area of town 3, then anchors on the anchored path are centers of town 0 and town 1. In this case, AGPF works as follows: the packet is first forwarded in the direction of the first anchor (the center of town 0). Once the packet arrives at some terminode that finds that the first anchor falls within its transmission range, the packet is then forwarded in the direction of the second anchor (the center of town 1). As before, when a packet comes to a terminode that is close to the second anchor, the packet is then forwarded in the direction of D's location.

Assuming that there are terminodes to ensure dense network connectivity in town areas and on highways, the packet is forwarded with AGPF mostly in the $greedy$ mode: packets are forwarded to terminodes that are always progressively closer to an anchor point or the destination. If however, occasionally there are regions of the network where such a greedy path does not exist, the packet is forwarded in the *perimeter mode* [39]: the packet successively traverses closer faces on a planar subgraph of the full network connectivity graph, until it reaches a node closer to an anchor or the destination and then greedy forwarding is resumed. In our implementation, a planar subgraph is the Gabriel subgraph [23] of the original full network connectivity graph. The perimeter mode packet forwarding is implemented according to the algorithm given in Appendix B.

We illustrate packet forwarding with the combination of both GPF and AGPF, and when only GPF is used in the example presented in Figure 5.9. Here, S is in town 0 and D is town 3. In the case of AGPF, S sets the anchored path to consist of one anchor: center of town 1. AGPF forwards the packet along the path that goes to town 1. Once the packet is close to the center of town 1, the packet is forwarded towards D by using GPF. Packet forwarding is almost always in greedy mode; however, there are cases where the perimeter mode is used for a very few (2-3) hops, before greedy forwarding resumes.

GPF uses a much longer path across town 2. Figure 5.9 illustrates that the packet is first forwarded in the greedy mode toward D until it reaches terminode $P1$, where perimeter mode starts because $P1$ does not have a neighbour closer to D than itself. The packet is thus forwarded in perimeter mode until greedy mode resumes at node $G1$ ($G1$ that is closer to D than $P1$). At $P2$, packet forwarding starts again perimeter mode. In this mode, a packet is forwarded from town 2 back to town 0, and from there through town 1 and town 3. Finally, when the packet arrives at $G2$, which is closer to D then $P2$ (where perimeter mode is started), greedy mode resumes until the packet is received by D.

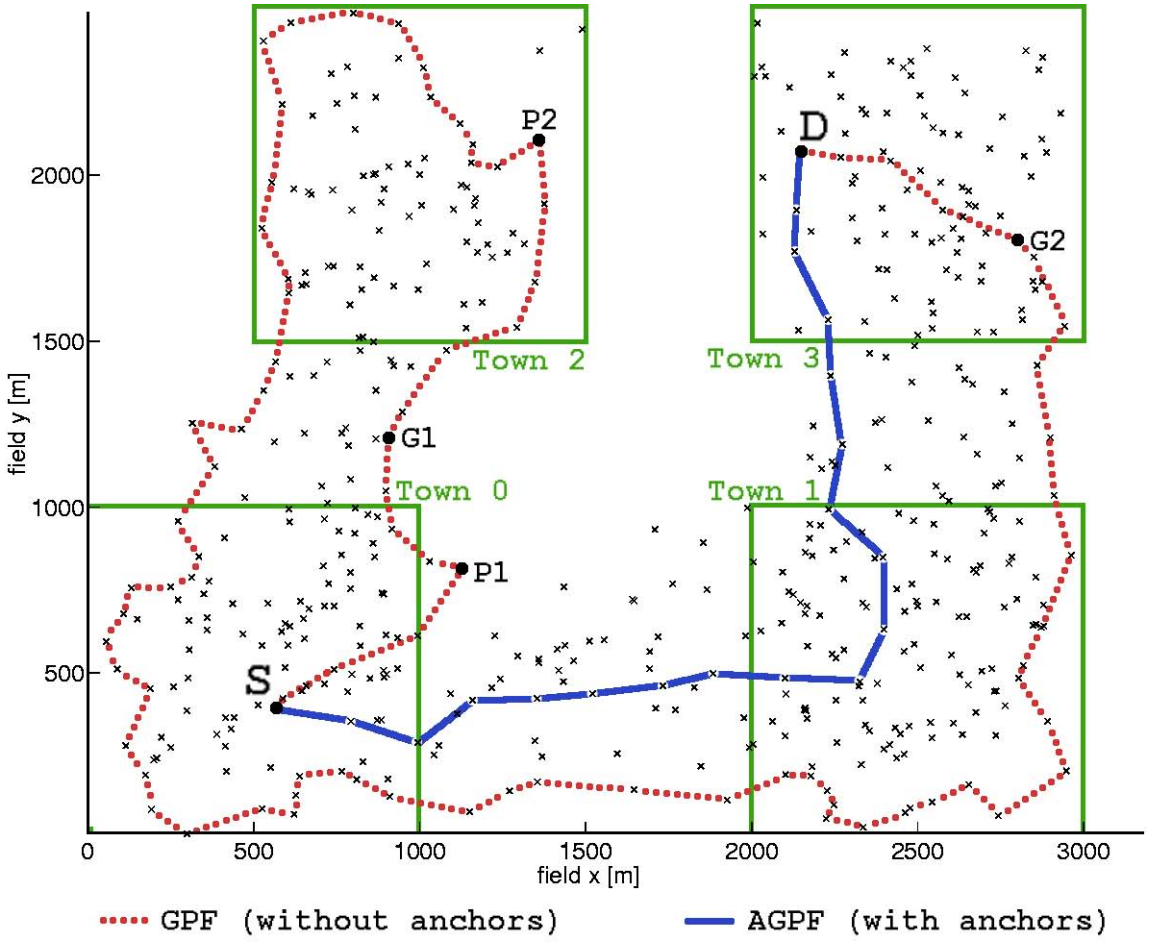


Figure 5.9: Figure presents the path of the packet from source S to destination D in case of two routing protocols: GPF that does not use anchors and AGPF with anchors. AGPF gives a shorter path than GPF.

Figure 5.9 clearly illustrates the case where the usage of anchors give shorter paths than when anchors are not used.

5.5.3 Simulation Parameters

In order to evaluate the benefits of using anchors, we conducted simulations of 500 terminodes forming an ad hoc network. The size of the simulated area is 3000m x 2500m.

Terminodes move between 4 towns inside the simulation area presented in Figure 5.8. The centers of the four towns have coordinates: (550 m, 550 m), (2500 m, 500 m), (1000 m, 2000 m) and (2500 m, 2000 m) respectively. The town area is a square around the town center with the

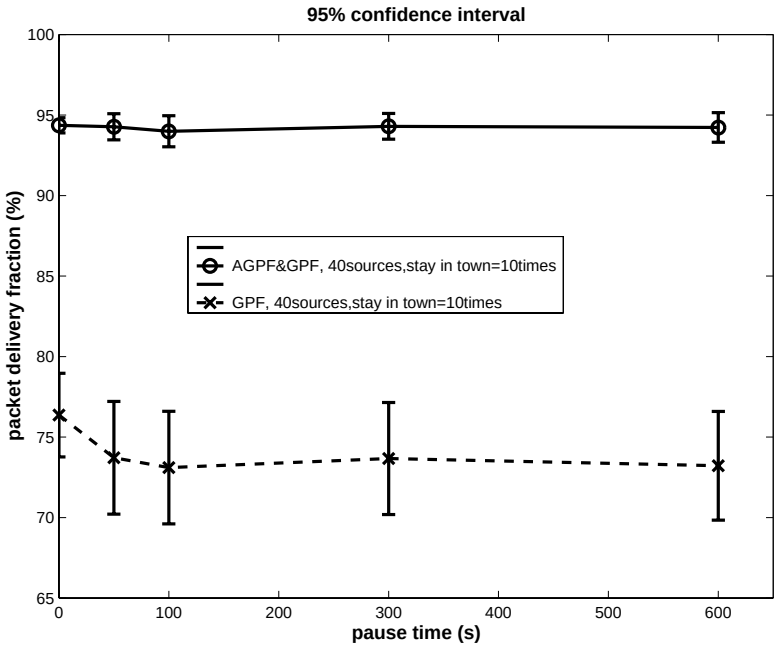
width of 500 m from the town center. There are 200 ordinary terminodes and 300 commuters.

The mobility model is the restricted waypoint mobility model. An ordinary terminode begins its journey from a random location inside random town. As described above, it moves *stay_in_town* times inside the same town and then selects another random town to move. For each movement, a terminode takes a random speed that is uniformly distributed between 0-20m/s; before each movement, a terminode pauses for some pause time. We ran simulations with different pause times and different *stay_in_town* parameters of ordinary terminodes. These parameters define different degrees of ordinary terminode mobility. A longer pause time means that ordinary terminodes are less mobile. For a fixed pause time, a larger *stay_in_town* means that a terminode is staying longer within a geographic region that corresponds to a single town. We consider different mobility rates of ordinary terminodes because this is the set of nodes where all traffic sources and destinations come from. In our simulations, commuters have higher mobility than ordinary terminodes. For their movements they take a random speed that is uniformly distributed between 0-20m/s. The pause time for commuter terminodes is equal to 1 second; once they reach a town, another random destination inside a different town is chosen for the subsequent movement.

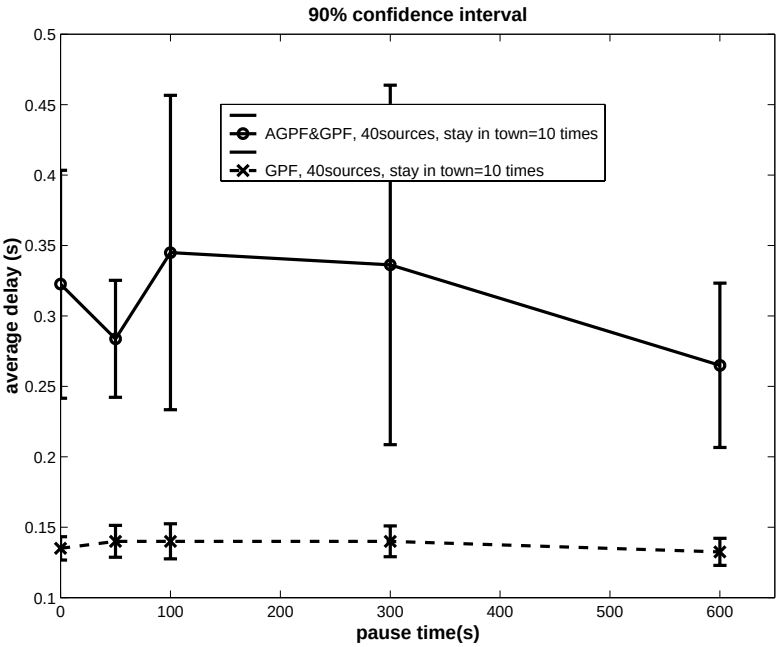
The location information lifetime parameter is set to 5 seconds.

Traffic sources are continuous bit rate (CBR). The source-destination pairs are spread randomly over the network. All CBR sources send two packets per second, and uses 64-byte packets. All communication patterns are peer-to-peer. All source destination pairs are chosen from the group of ordinary terminodes. Recall that these terminodes simulate small personal devices that stay within a boundary of a single town for a number of movements. Commuters are fast moving nodes, which are introduced in simulations in order to ensure a connected network. Their role is to relay packet on behalf of ordinary terminodes.

We performed four sets of simulations with different node mobility degrees and traffic load. In the first two sets there are 40 CBR sources, in the third set of simulations there are 50 CBR sources, while in the fourth set there are 150 CBR sources. CBR connections are started at times uniformly distributed between 400 and 500 seconds (starting from initial positions at the beginning of the simulation this time is enough for terminodes to establish the network of towns and highways), and they last until the end of simulation. All simulations last for 1200 seconds. In these simulations the average number of hops that packets take is the interval between 8 and 12 hops.



(a)



(b)

Figure 5.10: (a) Packet Delivery Fraction with 40 sources; stay_in_town parameter is 10, (b) Average delay with 40 sources; stay_in_town parameter is 10

5.5.4 Simulation Results

We carried out a performance study of two routing protocols for the network in Figure 5.8. The first protocol is TRR with both elements: *GPF* and *AGPF*. The second protocol uses only GPF. Section 5.5.2 explains simulation scenarios where the two protocols are examined.

Below we present the simulation results for:

- packet delivery fraction - the ratio of the data packets delivered to the destinations to data packets generated by the CBR sources;
- average end-to-end delay of data packets - includes all possible delays: queuing at the interface queue, retransmission delays at the MAC, and the propagation and transmit times.

Recall that in our simulations we use the idealized location management with no overhead. In addition, GMPD that is used for anchored paths discovery assumes that the network map is known to all nodes with no overhead. Therefore, the only routing overhead in simulations is due to HELLO messages.

The state information that terminodes keep is also minimal: they keep only the information about their one and two hops neighbours (as we described in Section 4.4).

In all figures that present our simulation results each data point presents an average of at least six simulations with identical traffic models, but different randomly generated movement patterns. We evaluated the two protocols by varying mobility and traffic load levels. In order to examine the performance of the routing protocol under different degrees of congestion, we varied the number of CBR sources in the network.

Varying the $stay\_in\_town$ parameter

In the first set of simulations, the *stay_in_town* parameter is set to 10 for ordinary terminodes (CBR sources and sinks).

Different degrees of mobility are obtained for different pause times of ordinary terminodes. For higher pause times, because *stay_in_town* is high, ordinary terminodes for most of the simulation time move inside the same town area. For smaller pause times they move to different town areas more frequently. Figure 5.10 (a) shows that the combination of GPF and AGPF delivers about 20 percent more packets compared to the case where only GPF is used. This result is explained as follows. GPF can give complex and long paths for source-destination pairs that are situated in towns not connected with a highway (Figure 5.9). For those packets there is a higher probability that they will be dropped. Moreover, because GPF often uses perimeter-mode packet

forwarding, many packets are dropped due to loops. Because in this simulation CBR sources and destinations do not frequently change town areas, there are several flows where GPF loses many packets, while there AGPF has more success. We described this problem in Section 3.3.3.

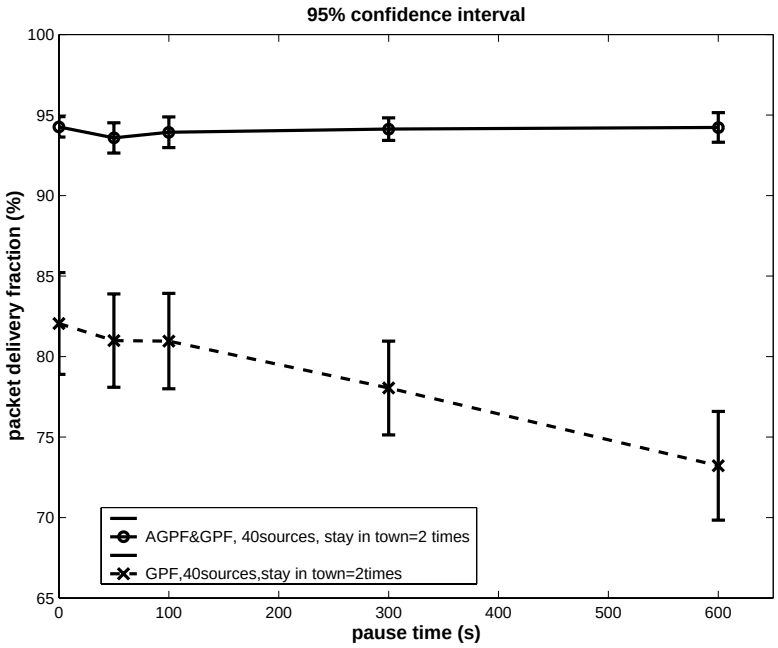
We may observe that for both protocols, the packet delivery fraction is stable for different pause time. There are two reasons for that: First, note that in our simulations, sources know destination positions accurately enough at all times. The number of packets that is dropped due to location inaccuracy is small even when the pause time is small. The main reason for packet dropping is either buffer overflow due to congestion or because of the perimeter-mode loop problem. Second, because we use the idealized location management with no overhead, the only routing load concerns sending of HELLO messages. Since every node periodically generates HELLO messages, the overhead due to HELLO messages is independent of the mobility rate and the traffic load. Therefore, the pause time parameter that influence the mobility rate does not have a big impact on packet delivery success.

Figure 5.10 (b) illustrates average end-to-end delays for the first set of simulations. GPF has smaller delay than AGPF. The reason is that GPF has a lower packet delivery fraction than AGPF and the average delay counts only for delivered packets. We observed that with GPF a large number of the packets that take long paths are dropped, and that most of the packets that are received at the destination experienced short paths, with short delays.

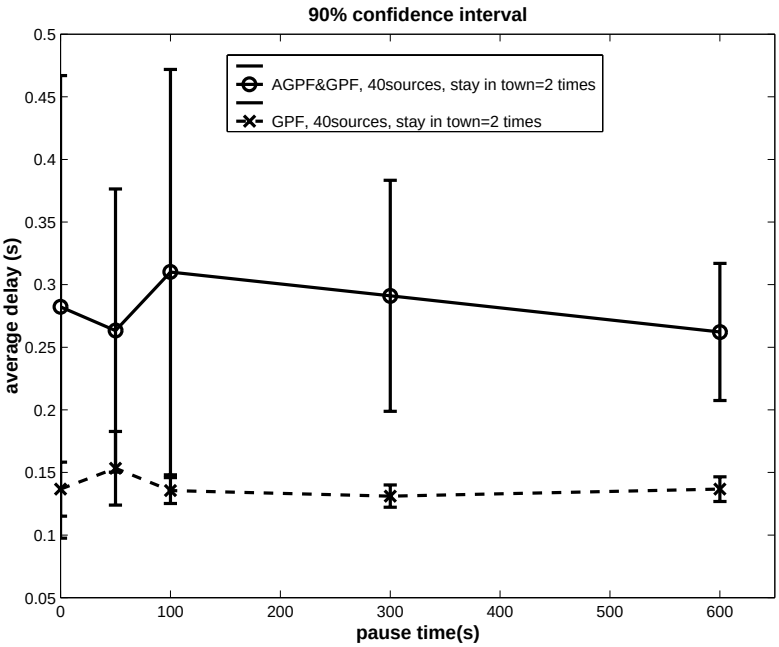
The results of the second set simulations are presented in Figures 5.11. All simulation parameters are the same as in the previous simulation, except that ordinary terminodes move more frequently from one town area to another. Here the *stay_in_town* parameter is set to 2. We observe that GPF delivers more packets than in the previous simulations (Figure 5.10). This can be explained: with increased mobility, those source-destination pairs for which GPF gives a small fraction delivery in the previous simulations can move to towns where GPF gives a better path. In this way bad situations, where GPF gives long complex paths, do not last for the duration of the simulation. This is especially true for lower pause times. For higher pause times, again we observe GPF decreases in the packet delivery fraction.

Varying the number of CBR sources

The results of the third set of simulations are presented in Figure 5.12. These simulations differ from the first set of simulations in that the number of CBR sources in the network is increased to 50. We observe that the combination of AGPF and GPF again delivers more packets than when only GPF is used. However, AGPF decreases its delivery fraction compared to the previous case when there are 40 CBR applications. The average delay is considerably increased compared to

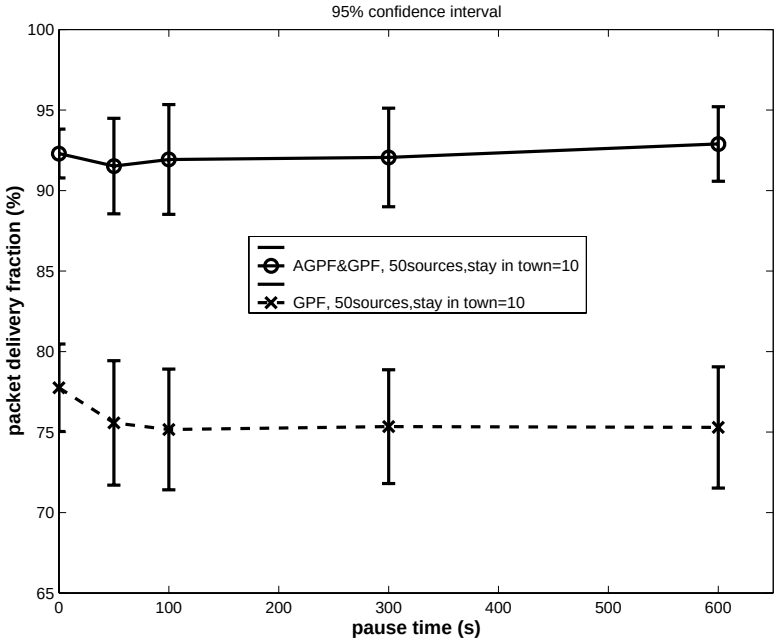


(a)

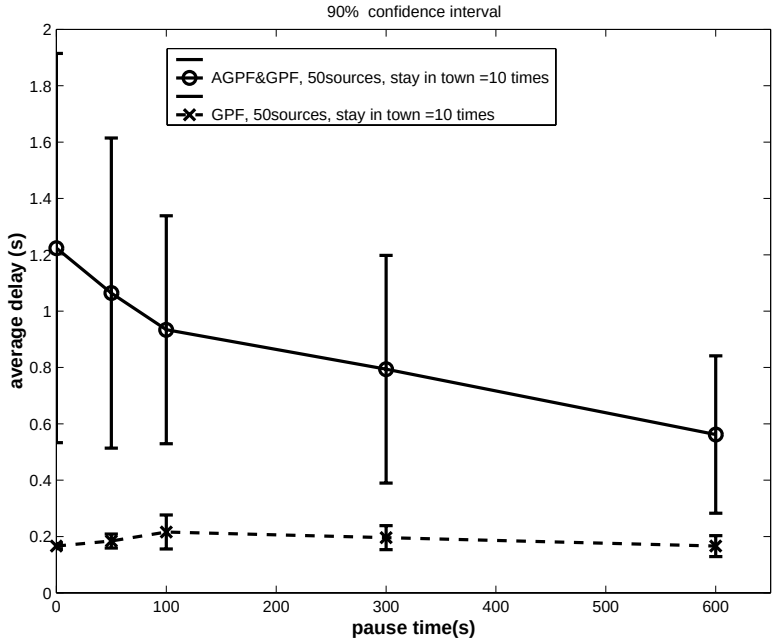


(b)

Figure 5.11: (a) Packet Delivery Fraction with 40 sources; stay_in_town parameter is 2, (b) Average delay with 40 sources; stay_in_town parameter is 2



(a)



(b)

Figure 5.12: (a) Packet Delivery Fraction with 50 sources; stay_in_town parameter is 10, (b) Average delay with 50 sources; stay_in_town parameter is 10

the case when there are 40 CBR applications. Indeed in this simulation, we observed an increased level of congestion in the region of the network between town 0 and town 1. Since AGPF directs most of its anchored paths across this region, that result in congestion and the increased number of dropped packets. GPF performs similarly as in the previous simulation with smaller number of sources. The explanation why GPF is less susceptible to the increased number of CBR sources than AGPF is the following. When the source town is town 0 and the destination town is town 3, AGPF forwards the packets via the region between town 0 and town 1. Unlike AGPF, GPF does not directly forward the packets in direction of town 1. Firstly, GPF forwards it to town 2 and than back to town 0 where the packet is forwarded in direction of town 1. Provided that the packet is not lost during the journey between towns 0 and 2, it contributes in the congestion along the highway between town 0 and town 1. However, in our simulations we have observed that there are many packets that are lost before taking this highway. Thus this explains why GPF is less susceptible to the increased number of CBR sources than AGPF.

This is an example where multipath routing for AGPF would be beneficial. In our simple network topology based on four towns, AGPF uses only one anchored path to the destination. If, however, there were several anchored paths over which packets can be sent, this would result in load balancing.

The fourth set of simulations is performed with 150 CBR sources. CBR applications are started at times uniformly distributed between 400 and 1000 seconds. For each connection two 64-byte data packets are sent per second for 40 seconds. In these simulations we have more connections than in the first and second set of simulations, however, here connections last for less time and the total number of packets is smaller. (We verified by simulation that when all 150 CBR connections last for the whole simulation time, this creates a high congestion in the network. Because our goal is to measure routing protocols behaviour, not the limitation of the IEEE 802.11 MAC for data packet capacity, we were not interested in this case.) Figure 5.13 (a) illustrates better delivery rate in the case when AGPF is used than without AGPF. Concerning the average end-to-end delay, Figure 5.13 (b) illustrates that a 90% confidence interval is much smaller than in the previous simulations. The reason is that the overall number of packets sent by CBR sources is much less than in the first and second set of simulations, and therefore less samples are taken onto account when end-to-end delay is determined.

We conclude that in all our simulations the combination of AGPF and GPF results in a higher packet delivery fraction than when only GPF is used. We observe that the improvement of AGPF over GPF is more important for a higher $stay\_in\_town$ parameter (nodes stay in single town areas for longer time).

Slow simulation speed and large memory requirements of GloMoSim prevented us from simulating larger networks. We believe that within a larger area, the benefits of good anchored paths over complex, long GPF paths will be even more evident.

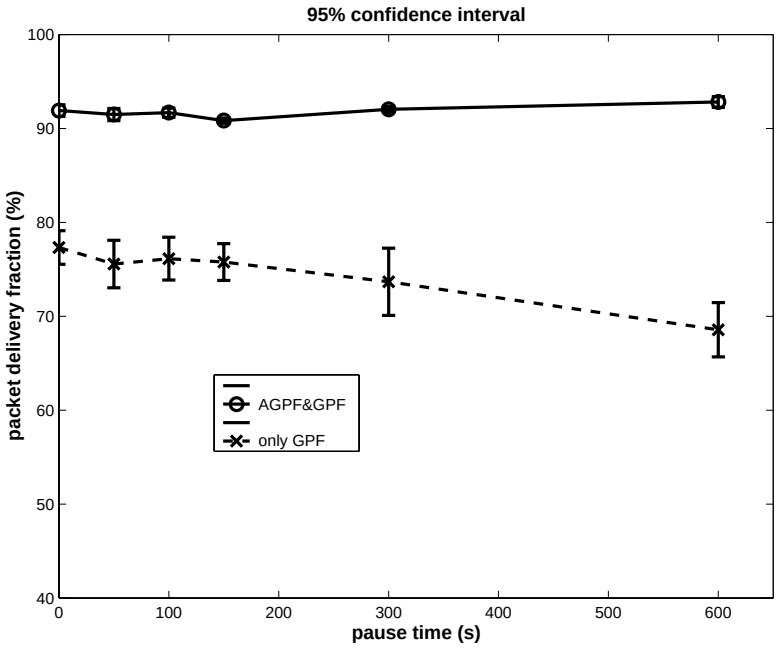
5.6 Conclusion

In this chapter we describe implementation of terminode routing in GloMoSim. We performed simulations in different sized mobile ad hoc networks.

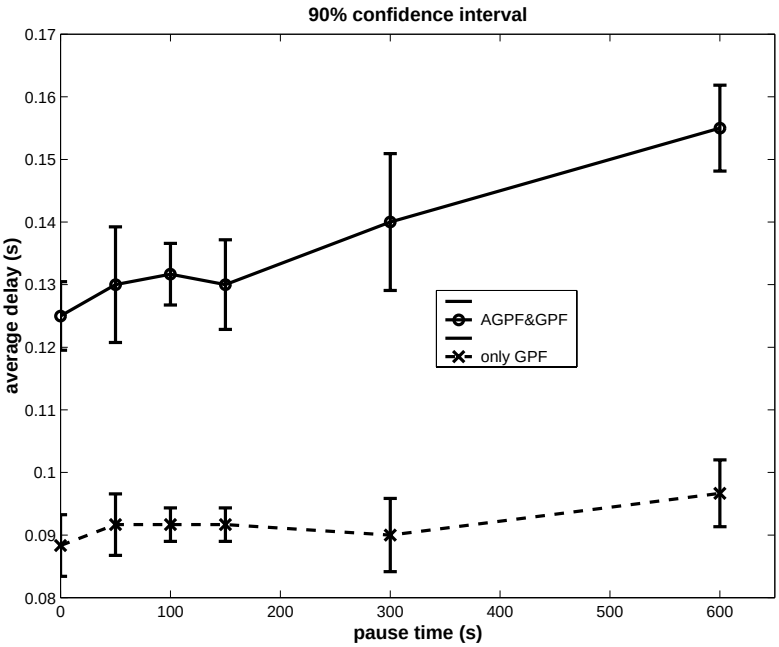
In small mobile ad hoc networks composed of 100 uniformly distributed nodes, terminode routing is compared to two other routing protocols, AODV and LAR1. Our simulation results indicate that terminode routing outperforms AODV and LAR1 in terms of packet delivery fraction and normalized routing load.

In a large network of 600 uniformly distributed nodes, we shown by simulations that terminode routing methods TLR and RLF help to alleviate problems due to destination location inaccuracy.

The last set of simulations was performed in a larger mobile ad hoc network composed of 500 nodes, which are not uniformly distributed. The aim of our simulations was to illustrate the benefits of the AGPF method and anchors for packet forwarding over GPF that only uses the destination location for packet forwarding. We introduced the topology based on four towns where nodes move between towns according to the mobility model that we called “restricted random waypoint”. Our simulation results demonstrate improvements in packet delivery fraction when the combination of AGPF and GPF is used, than when only GPF is used.



(a)



(b)

Figure 5.13: (a) Packet Delivery Fraction with 150 sources; stay_in_town parameter is 2, (b) Average delay with 150 sources; stay_in_town parameter is 2

Chapter 6

Future Work

In this chapter we give some directions of our future work.

6.1 Geodesic Medium Access Protocol (GEOMAC)

In Chapter 4, we presented the terminode routing protocol. There, we said that TRR is used for remote destinations (not-TLR-reachable destinations) and it uses geographical locations. For a given reference location (i.e., the destination location or an anchor), a node determines the next hop from its neighbour list such that the distance to the reference location is the most reduced. Nodes learn about their neighbours' identities and positions by periodically sending HELLO messages.

In order to maintain an accurate neighbour list, intervals of sending HELLO messages should be short (in our implementation of terminode routing, HELLO messages are sent every second, which we verified to ensure accurate neighbour lists in the case when nodes move with either a speed of a pedestrian or a car). Otherwise, if the list of neighbours is not updated on a regular basis, loops are possible.

One example of such a loop is described with the following scenario. Let's assume that node A has in its list of neighbours node B. However, the actual B's location is different from the one that A has. Assume next, that A has a packet to send towards location LOC. A selects as the next hop to send the packet, node B, because A thinks that B is closest to LOC. If B is still in A's transmission range it receives the packet. The actual B location is such that the distance to LOC is not smaller than the one from A to LOC. Once B receives the packet, it finds that its neighbour A is closer to LOC and B sends the packet back to A.

The described loop scenario is very probable in the case when nodes are moving extremely

fast and the mechanism of HELLO-ing is not enough to track immediate neighbours.

One possible solution to alleviate loops problem consists in all data packets carrying their local sender's position. In the previous loop scenario, when B sends back the packet to A, A updates B's actual location, and the packet will not be sent back to B. This solution would work in regions under traffic load.

We propose a new MAC protocol, that is called geodesic MAC (GEOMAC). GEOMAC is the modification of the standard IEEE 802.11 MAC protocol [15] in order to combat the looping problem due to inaccuracy of neighbours locations.

GEOMAC is intended to be used in conjunction with position-based routing, such is terminode routing. The difference is that the next hop is resolved at the MAC layer, and not at the network layer. The basic idea behind GEOMAC is illustrated in Figure 6.1.

In GEOMAC, the RTS frame includes two location information: the physical location of the RTS frame sender and the reference location LOC (the physical location of destination, or an anchor). Again, let's assume that node A has a data packet to send in the direction of LOC. Then, node A broadcasts the RTS frame. However, unlike IEEE 802.11 MAC protocol, the RTS does not carry the intended receiver. Several nodes receive the RTS frame. However, it is required that only one node, which has a lower distance to LOC, responds with CTS. In GEOMAC, nodes that detect that they are closer to LOC than the sender of the RTS frame, wait a random time. When they receive the RTS frame, the $send\_CTS$ timer is set. If host X receives the CTS frame before its own $send\_CTS$ timer expires, then X defers sending the CTS frame. Otherwise, X sends the CTS frame. If A receives the CTS frame successfully, from some node X, it starts sending the data frame to X.

There are several possible solutions for the interval over which the $send\_CTS$ timer is set. One approach is that this interval is proportional to the distance to the destination. The closer to the destination the node is, the shorter the interval of the $send\_CTS$ timer is.

The implementation and simulation of GEOMAC is the subject of our ongoing work.

6.2 Other Issues

There is a number of issues that we mentioned in this thesis that are matter of our future work. These issues are listed below.

- *Friend Assisted Path Discovery (FAPD):* In Section 4.9.1 we described FAPD. Its implementation in the GloMoSim simulator and testing are left to be done.

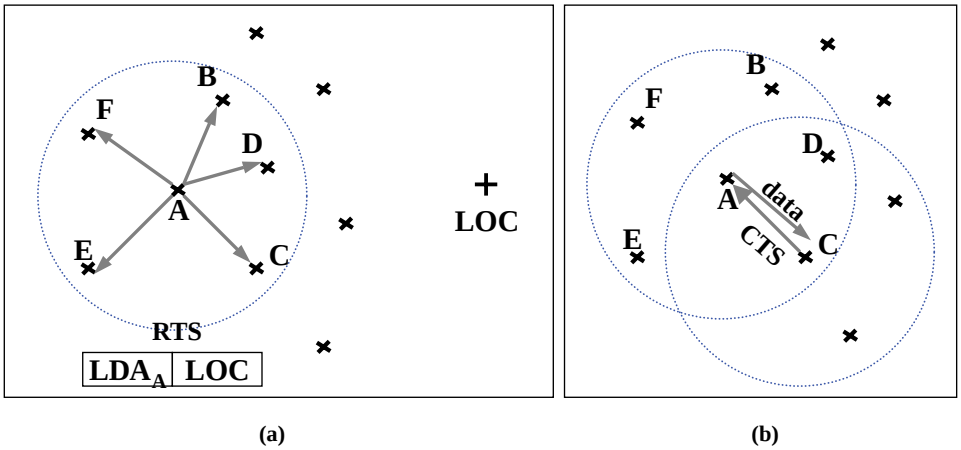


Figure 6.1: Figure presents illustration of GEOMAC. (a): Node A that has to forward the packet in direction of LOC, broadcasts the RTS frame. The RTS frame includes the location of A (LDA_A) and LOC. Neighbouring nodes B, C, D, E, F receive the RTS frame. Nodes B, C, D, because closer to LOC than A, set $send\_CTS$ timers to a random value. (b): The first timer that expires is the one of C. C broadcasts CTS to A. Node D that receives CTS from C stops its $send\_CTS$ timer and defers sending CTS. As soon as A receives CTS from C, it enters into transmitting mode and sends the data frame to C.

- *path maintenance:* In Section 4.12.2 we gave a short description of how sources maintain acquired paths. Further analysis, implementation and testing of the path maintenance are left for future work.

- *multipath terminode routing:* In Section 4.12 we gave the architecture of how to apply multipath terminode routing. Implementation and analysis of multipath terminode routing is also for future work.

- *location service:* In Section 4.11 we gave a description of the VHR location service, which is suitable for applying within a terminode network. It is left for future work to integrate the VHR location service and terminode routing in GloMoSim environment and to test how the two work together.

Chapter 7

Conclusion

In this thesis, we studied routing protocols with applications to mobility.

In Chapter 2, we considered the problem of scalable multicast routing in a large single domain network with specific requirements that there is a large number of small multicast groups. First, we gave the review of the existing multicast routing protocols. We explained why the existing protocols do not scale well in the scenario that we are interested in. Our solution to the problem was the design of the DCM routing protocol. DCM is based on an extension of the centre-based tree approach. DCM uses several core routers, called DCRs and a special control protocol among them. The objectives achieved with DCM are: (1) avoiding state information in backbone routers, (2) avoiding triangular routing across expensive backbone links, (3) scaling well with the number of multicast groups. Our results indicated that DCM performs better than the existing sparse mode routing protocols in terms of multicast forwarding table size. We gave examples of the application of DCM: in distributed simulations; in supporting Internet host mobility, and in cellular Internet Telephony.

The second problem addressed in this thesis is scalable routing in a large mobile ad hoc network.

Chapter 3 is an overview of the existing routing protocols for mobile ad hoc networks. We gave an overview of traditional MANET routing protocols, as well as an overview of position-based routing protocols. We explained why position-based routing protocols are more scalable for large mobile ad hoc networks than the traditional ones.

Chapter 4 contains the description of a scalable routing protocol for a large mobile ad hoc network that is called terminode network. We call the nodes terminodes because they act as network nodes and terminals at the same time. Our routing scheme is the combination of the two protocols, TLR and TRR. The use of TRR results in a scalable solution that reduces de-

pendence on the intermediate systems, while TLR allows to increase the probability of reaching the destination even when it has moved considerably from the the location known at the source. In order to circumvent large holes in terminode distribution, TRR primarily forwards packets on *anchored paths*. In contrast with traditional routing algorithms, an anchored path does not consist of a list of nodes to be visited to reach the destination. An anchored path is a list of fixed geographic points, called $anchors$. In order to forward packets along an anchored path, TRR uses the method AGPF. In a highly mobile ad hoc network, paths that once worked well can easily be broken or become congested. As a response to such uncertainty in the network, we advocate that routing in a large self-organized ad hoc network should be multipath. We give the outline of how to perform multipath routing in a terminode network.

Terminode routing was implemented in GloMoSim and Chapter 5 contains the terminode routing evaluation by using simulations. Both small and large mobile ad hoc networks were considered. In small mobile ad hoc networks composed of 100 uniformly distributed nodes, terminode routing is compared to two other routing protocols, AODV and LAR1. Our simulation results indicate that terminode routing outperforms AODV and LAR1 in terms of packet delivery fraction and normalized routing load. In a large network of up to 600 uniformly distributed nodes, we shown by simulations that terminode routing methods TLR and RLF help to alleviate problems due to location inaccuracy. The last set of simulations was performed in a larger mobile ad hoc network composed of 500 nodes, which are not uniformly distributed. The aim of our simulations was to illustrate the benefits of the AGPF method for packet forwarding. We introduced the realistic topology based on towns and highways where nodes move between towns according to the mobility model that we called “restricted random waypoint”. Our simulation results demonstrate that the using of anchored paths and AGPF improve packet delivery fraction compared to when anchors are not used.

Finally, Chapter 6 describes some avenues for further extension of the work presented in this thesis.

Bibliography

[1] Network Simulator. Available from http://www-mash.cs.berkeley.edu/ns.

[2] www.terminodes.org.

[3] Hierarchical Protocol Independent Multicast (HPIM). Available from www.cs.ucl.ac.uk/staff/jon/mmbook/book/book.html, 1997.

[4] R. Bagrodia, R. Meyer, M. Takai, Y. Chen, X. Zeng, J. Martin, and H.Y. Song. PARSEC: A Parallel Simulation Environment for Complex Systems. *IEEE Computer*, 31(10), October 1998.

[5] A. Ballardie. Core Based Trees (CBT) Multicast Routing Architecture. RFC 2201, September 1997.

[6] L. Barriere, P. Fraigniaud, L. Narajanan, and J. Opatrny. Robust position based routing in wireless ad hoc networks with unstable transmission ranges. *5th Int. Workshop on Descrete Algorithms and methods for mobile computing communications (DIAL M)*, July 2001.

[7] S. Basagni, I. Chlamtac, V. Syrotiuk, and B. Woodward. A Distance Routing Effect Algorithm for Mobility (DREAM). *Proceedings of the Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom'98), Dallas, Texas, USA*, August 1998.

[8] T. Bates. Multiprotocol Extensions to BGP-4. IETF RFC 2283, 1998.

[9] R. Boivie. A New Multicast Scheme for Small Groups. IBM Research Report, June, 1999.

[10] P. Bose, L. Devroye, W. Evans, and D. Kirkpatrick. On the spanning ratio of gabriel and β-sceleton. *submitted to SIAM Journal on Discrete Mathematics*, 2001.

[11] P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia. Routing with guaranteed delivery in ad hoc wireless newtorks. *3rd Int. Workshop on Descrete Algorithms and methods for mobile computing communications (DIAL M)*, August 1999.

[12] J. Broch, D.A. Maltz, D.B. Johnson, Y.C Hu, and J. Jetcheva. A performance comparison of multi-hop wireless ad hoc network routing protocols. *Proceedings of the Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom'98), Dallas, Texas, USA*, August 1998.

[13] Tuch Bruce. Development of WaveLAN, an ISM Band Wireless LAN. *AT&T Technical Journal*, July/August 1993.

[14] Srdjan Capkun, Maher Hamdi, and Jean-Pierre Hubaux. GPS-free positioning in mobile Ad-Hoc networks. *Proceedings of the 34th HICSS*, January 2001.

[15] IEEE Computer Society LAN MAN Standards Committee. Wireless LAN Medium Access Protocol (MAC) and Physical Layer (PHY) Specification. *IEEE Std 802.11-1997, The Institute of Electrical and Electronics Engineers*, New York, 1997.

[16] Deborah Estrin, Mark Handley, Ahmed Helmy, Polly Huang, and David Thaler. A Dynamic Mechanism for Rendezvous-based Multicast Routing. In *Proc. of IEEE INFOCOM'99*, New York, USA, March 1999.

[17] D. Estrin et.al. Protocol Independent Multicast-Sparse Mode (PIM-SM): Protocol Specification. RFC 2117, June 1997.

[18] D. Farinacci et. al. Multicast Source Discovery Protocol (MSDP). Internet Draft(work in Progress), June 1998.

[19] W. Fenner. Internet Group Management Protocol, Version 2. RFC 2236, November 1997.

[20] B. Fenner et. al. Multicast Source Discovery protocol MIB. Internet Draft(work in Progress), May 1999.

[21] G. Finn. Routing and addressing problems in large metropolitan-scale internetworks. *ISI Research Report ISU/RR-87-180*, March, 1987.

[22] M. Gerla G. Pei and X. Hong. Lanmar: Landmark routing for large scale wireless ad hoc networks with group mobility. In *Proceedings of the First IEEE/ACM Workshop on Mobile Ad Hoc Networking and Computing (MobiHOC)*, August 2000.

[23] K. Gabriel and R. Sokal. A new statistical approach to geographic variation analysis. *Systematic Zoology*, 18, 1969.

[24] J. Gao, L. Guibas, J. Hershberger, L. Zhang, and An Zhu. Geometric spanner for routing in mobile networks. In *Proceedings of the Second IEEE/ACM Workshop on Mobile Ad Hoc Networking and Computing (MobiHOC)*, October 2001.

[25] F. Glover. Future paths for integer programming and links to artificial intelligence. *Computers and Operational Research*, 13, 1986.

[26] C. Graff. IPv4 Option for Sender Directed Multi-Destination Delivery. RFC 1770, 1995.

[27] M. Handley and H. Schulzrinne et. al. SIP: Session Initiation Protocol. RFC 2543, 1999.

[28] P. Hansen. The steepest ascent mildest descent heuristic for combinatorial programming. *Proc. Congr. on Numerical Method in Combinatorial Programming*, Academic Press, 1986.

[29] A. Harter and A. Hopper. A new location technique for the active office. *IEEE Personal Communications*, pages 42–47, October 1997.

[30] H. Holbrook and D. Cheriton. IP multicast Channels: EXPRESS Support for Large-scale Single source Applications. In *Proc.of ACM SIGCOMM'99*.

[31] T.-C. Hou and V. Li. Transmission Range Control in Multihop Packet Radio Networks. *IEEE Transactions on Communications, Vol.Com-34, No.1*, January 1986.

[32] John Ioannidis, Dan Duchamp, and Gerald Q. Maguire Jr. IP-based Protocols for Mobile Internetworking. In *Proc. of SIGCOMM'91*, Zurich, Switzerland, September 1991.

[33] A. Iwata, C.-C. Chiang, G. Pei, M. Gerla, and Chen T.-W. Scalable Routing Strategies for Ad-hoc Wireless Networks. *IEEE JSAC*, 17(8), August 1999.

[34] P. Jacquet, P. Muhlenthaler, and A. Qayyum. Optimized Link State Routing Protocol. *Internet Draft, draft-ietf-manet-olsr-*.txt. Work in progress*.

[35] M. Jiang, J. Li, and Y. Tay. Cluster Based Routing Protocol(CBRP) Functional Specification. Internet Draft(work in Progress), August 1998.

[36] M. Joa-Ng and I. Lu. A peer-to-peer zone-based two-level link state routing for mobile ad hoc networks. *IEEE Journal of Selected Areas in Communications*, 17(8), 1999.

[37] D.B. Johnson and D.A. Maltz. Dynamic source routing in ad hoc wireless networks. in Mobile Computing, T. Imielinski and H. Korth, Eds. Norwell , MA: Kluwer, ch. 5, pp 153-181, 1996.

[38] Brad Karp. Geographic Routing for Wireless Networks, PhD Thesis. *Harvard University*, October 2000.

[39] Brad Karp and H.T. Kung. GPSR: Greedy Perimeter Stateless Routing for Wireless Networks. *Proceedings of ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom'00)*, August 2000.

[40] R.H. Katz, J.M. Kahn, and K.S.J. Pister. Mobile networking for 'smart dust'. *Proceedings of the Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom'99)*, August 1999.

[41] F. Kelly, S. Maulloo, and D. Tan. Rate control for communication networks: Shadow prices, proportional fairness and stability. *Journal of Operations Research Society*, March 1998.

[42] Frank Kelly. Mathematical modelling of the Internet . In *Proc. of Fourth International Congress on Industrial and Applied Mathematics*, 1999.

[43] Jon Kleinberg. The small-world phenomenon: an algorithmic perspective. *Technical Report 99-1776, Cornell Computer Science*, 1999.

[44] E. Kranakis, H. Singh, and J. Urrutia. Compass routing on geometric networks. *Proceedings of 11th Canadian Conference on Computational Geometry*, August 1999.

[45] S. Low and D. Lapsley. Optimization flow control, I:basic algorithm and convergence. *IEEE/ACM Transactions on Networking*, December 1999.

[46] S. Low and D. Lapsley. Random Early Marking for Internet Congestion Control. In *Proc. of IEEE Globecom'99*, December, 1999.

[47] S. Low and D. Lapsley. Optimization flow control with on-line measurement. In *Proc. of ITC*, volume 16, June, 1999.

[48] M. Lubi, M. Mitzenmacher, A. Shokrollahi, D. Spielman, and V. Stemann. Practical Loss-Resilient Codes. In *Proc. of the 29 ACM Symposium on Theory of Computing*, 1997.

[49] Michael Macedonia and Michael Zyda et.al. Exploiting Reality with Multicast Groups: A Network Architecture for Large-Scale Virtual Environments. In *IEEE Computer Graphics and Applications*, September 1995.

[50] J. Macker and M. S. Corson. Mobile As hoc Networking and the IETF. *ACM Mobile Computing and Communications Review*, 2(1), January 1998.

[51] J. Moy. Multicast Extensions to OSPF. RFC 1584, 1994.

[52] J. Moy. OSPF version 2. RFC 1583, 1994.

[53] S. Murthy and J.J. Garcia-Luna-Aveces. An efficient Routing Protocol for Wireless Networks. *ACM Mobile Networks and App. Journal, Special Issue on Routing in Mobile Communication Networks*, October 1996.

[54] Jayanth Mysore and Vaduvur Bharghavan. A New Multicasting-based Architecture for Internet Host Mobility. In *The Third Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom'97)*.

[55] R. Nelson and L. Kleinrock. The spacial capacity of a slotted ALOHA multihop packet radio network with capture. *IEEE Transactions on Communications*, 32(6), 1984.

[56] V. Park and M. S. Corson. A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Network. In *Proc. of IEEE INFOCOM'97*, Kobe, Japan, 1997.

[57] C. Perkins. IP Mobility Support, Network Working Group. RFC 2002, October 1996.

[58] C. Perkins. Minimal Encapsulation within IP. RFC 2004, 1996.

[59] C.E. Perkins and E.M. Royer. Ad-Hoc On-Demand Distance Vector Routing. *Proceedings of IEEE WMCSA'99, New Orleans*, Feb. 1999.

[60] Charles E. Perkins and Pravin Bhagwat. Highly dynamic destination-sequenced distance-vector (dsdv) routing for mobile computers. In *Proceedings of ACM SIGCOMM'94. London, UK.*, August 1994.

[61] Charles E. Perkins and David B. Johnson. Mobility Support in IPv6. In *Proc. of the Second Annual International Conference on Mobile Computing and Networking (MobiCom'96)*.

[62] Marc R. Perlman and Zygmunt J. Haas. Determining the Optimal Configuration for the Zone Routing Protocol. *IEEE JSAC*, 17(8), August 1999.

[63] M.R. Perlman, Z.J. Haas, P. Sholander, and S.S. Tabrizi. On the impact of alternate path routing for load balancing in mobile ad hoc networks. In *Proceedings of the First IEEE/ACM Workshop on Mobile Ad Hoc Networking and Computing (MobiHOC)*, August 2000.

[64] R. Perlman et.al. Simple Multicast: A Design for Simple, Low-Overhead Multicast. Internet Draft(work in Progress), February 1999.

[65] N. Priyantha, A. Chakraborty, and H. Balakrishnan. The Cricket location-support system. *Mobicom'00, Boston*, 2000.

[66] R. Jain, A. Puri , R. Sengupta. Geographical Routing Using Partial Information for Wireless Ad Hoc Networks. *IEEE INFOCOM*, 2001.

[67] S. Das, C. Perkins , E. Royer. Performance Comparison of Two On-demand Routing Protocols for Ad Hoc Networks. *IEEE INFOCOM*, 2000.

[68] S. Giordano, M. Hamdi. Mobility Management: The Virtual Home Region. Technical report no. ssc/1999/037, EPFL-ICA, 1999.

[69] Srinivasan Seshan, Hari Balakrishnan, and Randy H. Katz. Handoffs in Cellular Wireless Networks: The Daedalus Implementation and Experience. *Kluwer International Journal on Wireless Personal Communications*, January 1997.

[70] P. Sinha, R. Sivakumar, and V. Bharghavan. CEDAR: a Core-Extraction Distributed Ad hoc Routing Protocol. *IEEE INFOCOM*, March 1999.

[71] I. Stojmenovic and X. Lin. GEDIR: Loop-free location based routing in wireless networks. *Proceedings of the IASTED Int. Conf. on Parallel and Distributed Computing and Systems*, Boston, USA, November 1999.

[72] I. Stojmenovic and X. Lin. Loop-free hybrid single-path/flooding routing algorithms with guaranteed delivery for wireless networks. *IEEE Transactions on Parallel and Distributed Systems*, to appear.

[73] H. Takagi and L. Kleinrock. Optimal Transmission Ranges for Randomly Distributed Packet Radio Terminals. *IEEE Transactions on Communications, Vol.Com-32, No.3*, March 1984.

[74] M. Takai, L. Bajaj, R. Ahuja, R. Bagrodia, and M. Gerla. GloMoSim:A Scalable Network Simulation Environment. *Technical Report 990027, UCLA, Computer Science Department*, 1999.

[75] D. G. Thaler and C. V. Ravishankar. Using Name-Based Mappings to Increase Hit Rates. *IEEE/ACM Transactions on Networking*, 6(1), February 1998.

[76] David G. Thaler and Chinya V. Ravishankar. Distributed Center-Location Algorithms. *IEEE JSAC*, 15(3), April 1997.

[77] J. Tian and G. Neufeld. Forwarding State Reduction for Sparse Mode Multicast Communication. In *Proc. of IEEE INFOCOM'98*, March/April 1998.

[78] G. Toussaint. The relative neighbourhood graph of a finite planar set. *Pattern Recognition*, 12, 1980.

[79] P.F. Tsuchiya. The Landmark Hierarchy: A New Hierarchy for Routing in Very Large Networks. *Computer Communications Review*, 1988.

[80] USCG Navigation Center GPS page, May 2001. http://www.navcen.uscg.gov/gps/default.htm.

[81] V. W.-S. Wong, V. C.M. Leung. Location Management for Next-Generation Personnnal Communications Networks. *IEEE Network*, September/October 2000.

[82] Andras G. Valko. Cellular IP: A New Approach to Internet Host Mobility. *ACM SIGCOMM Computer Communicastion Review*, January 1999.

[83] Vinod Valloppillil and Keith W. Ross. Cache Array Routing Protocol v1.0. Internet Draft(work in Progress), 1998.

[84] D. Waitzman, S. Deering, and C. Partridge. Distance vector multicast routing protocol. RFC 1075, 1988.

[85] D. J. Watts. In *Small Worlds, The dynamics of networks between order and randomness*. Princeton University Press, 1999.

[86] Bernard M. Waxman. Routing of Multipoint connections. *IEEE JSAC*, 6(9), December 1988.

[87] Liming Wei and Deborah Estrin. The Trade-offs of Multicast Trees and Algorithms. In *Proc.of the 1994 International Conference on Computer Communications and Networks*, San Francisco, CA, USA, September 1994.

[88] P. Whittle. Optimization Under Constraints. *Wiley, Chichester*, 1971.

[89] Pawei Winter. Steiner problem in networks: A survey. *Networks*, 17(2), 1987.

[90] S.-C. Woo and S. Singh. Scalable Routing in Ad Hoc Networks. *Technical Report TR00.001, Dpt. of Electrical and Computer Engineering, Oregon State University, Corvallis, OR*, March 2000.

[91] S. Xu, S. Papavassiliou, and K. Amouris. On the Performance of a Scalable Single-Tier Position Based Routing Protocol for Mobile Ad-Hoc Wireless Networks. In *Proc. of ISCC*, 2000.

[92] Ko Y. and Vaidya N. Location-aided routing in mobile ad-hoc networks. *Proceedings of the Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom'98), Dallas, Texas, USA*, August 1998.

[93] Li. Yunzhou. Group Specific MSDP Peering. Internet Draft(work in Progress), June 1999.

Appendix A

The Optimal Routing Problem

In this appendix we present the theoretical work on optimal routing in a terminode network. We look at the problem of dynamic routing with an objective to maximize the global network utility. We show that multipath forwarding ensures that the utility of the network approaches optimum and that the optimal path is a multipath.

The theoretical results presented in this appendix are used in Section 4.12 to outline how to apply multipath routing in a terminode network.

A.1 Optimal Routing in a Terminode Network

For the purpose of this work on the optimal routing, a terminode network is presented, in a macroscopic way, as being composed of a number of blocks populated by terminodes. We assume that the spatial distribution of terminodes in each block follows the process that is known (e.g. Poisson point process). In addition, given the communication model inside the block, traffic patterns can be estimated through every block. Depending on the density of terminodes, traffic patterns in a block, and some additional information, we can characterize the transit capability of a block. Thus, the macroscopic presentation of a block is given by its transit capability (capacity). We assume that every terminode has a macroscopic view of all blocks and their transit capabilities.

Figure A.1 shows an example of a macroscopic presentation of a terminode network. The constituting blocks of the network are shown. The blocks colored in black are those that cannot be used for packet relaying (e.g. obstacles or blocks without terminodes).

We define the global utility $U(\cdot)$ as the sum over the user utilities minus the cost imposed by the network [42], under the constraints imposed by the transit capabilities of the blocks that

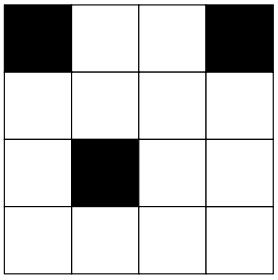


Figure A.1: An example of a macroscopic presentation of a terminode network. The constituting blocks are shown. The blocks colored in black are those that can not be used for relaying packets.

constitute a terminode network.

$$U(x) = \sum_{s \in \mathcal{S}} U_s(x_s) - \sum_{b \in \mathcal{B}} C_b(f_b), \tag{A.1}$$

where $x = (x_s, s \in \mathcal{S})$ is a vector of source rates, and $\mathcal{S}$, $\mathcal{B}$ are defined to be sets of the sources and blocks, respectively. Source $s \in \mathcal{S}$ is characterized by a utility function $U_s(\cdot)$ that is assumed to be a concave increasing in its transmission rate x_s. For instance, one may consider sources to be controlled by a TCP-like rate control algorithm, and define $U_s(\cdot)$ appropriately. The total cost of the network, the right-most term in Equation (A.1), is given by the sum of the costs of each block C_b. The latter, C_b, is a function of the amount of traffic f_b that traverses the given block b.

We look at the routing as an optimization problem, and we want to jointly optimize flow control and routing, such that the global utility function defined by Equation (A.1) is maximized. Global utility is optimized under the constraints imposed by the transit capabilities of the blocks that constitute a terminode network. We present below the model to find the source rates and routes that are used to send data from source to destination such that we get optimum of the global utility function.

Relevant work done on optimization based flow control can be found in [45, 46, 47, 42, 41].

A.2 The model for solving the optimal routing problem

Consider a network with a set $\mathcal{B}$ of resources. In our case, $\mathcal{B}$ is a set of blocks. Let a route r be a non-empty subset of $\mathcal{B}$, and write $\mathcal{R}$ for the set of possible routes in the network. Set $A_{br} = 1$ if $b \in r$, so that resource b lies on route r, and set $A_{br} = 0$ otherwise. This defines a

0-1 matrix $A = (A_{br}, b \in \mathcal{B}, r \in \mathcal{R})$. Suppose that several routes through the network may serve the same source-destination pair. The network is shared by a set $\mathcal{S}$ of source-destination pairs. A source-destination pair *s* can be served by a subset of $\mathcal{R}$ routes. Set $H_{sr} = 1$ if $r \in \mathcal{R}$, so that route r serves the source-destination s, and set $H_{sr} = 0$ otherwise. This defines a 0-1 matrix $H = (H_{sr}, s \in \mathcal{S}, r \in \mathcal{R})$. For each $r \in R$ let $s(r)$ identify a value $s \in \mathcal{S}$ such that $H_{sr} = 1$, and suppose this value is unique; we view $s(r)$ as the source-destination pair served by route *r*.

We associate a source-destination *s* with a user, and suppose that if a rate x_s is allocated to the source-destination *s* then this has utility $U_s(x_s)$ to the user. We assume that the utility $U_s(x_s)$ is an increasing, strictly concave and continuously differentiable function of x_s over the range $x_s \geq 0$. We further assume that utilities are additive, so that the aggregate utility of rates $x = (x_s, s \in \mathcal{S})$ is $\sum_{s \in \mathcal{S}} U_s(x_s)$.

Now let y_r be the flow on route r. Let's suppose that block *b* incurs a cost $C_b(\sum_{r:b \in r} y_r) = C_b(\sum_r A_{br} y_r)$ dependent on the flow through that block, where $C_b(\cdot)$ is a strictly convex and differentiable function. A flow pattern $y = (y_r, r \in \mathcal{R})$ supports the rates $x = (x_s, s \in \mathcal{S})$, so that the flows y_r over routes *r* serving the source-destination *s* sum to the rate x_s. Consider the following optimization problem:

$$\begin{aligned} \text{maximize} \quad & U(x) = \sum_{s \in \mathcal{S}} U_s(x_s) - \sum_{b \in \mathcal{B}} C_b(\sum_{r:b \in r} y_r) && \text{(A.2)} \\ \text{subject to} \quad & Hy = x && \text{(A.3)} \\ \text{over} \quad & x, y \geq 0 && \text{(A.4)} \end{aligned}$$

The objective function (A.2) is differentiable and strictly concave and the feasible region (A.3),(A.4) is compact; hence a maximizing value of (x, y) exists and can be found by Langrangian methods [88].

Consider the Langrangian form

$$\begin{aligned} L(x, y; \lambda) &= \sum_{s \in \mathcal{S}} U_s(x_s) - \sum_{b \in \mathcal{B}} C_b(\sum_{r:b \in r} y_r) - \lambda^T (x - Hy) \\ &= \sum_{s \in \mathcal{S}} (U_s(x_s) - \lambda_s x_s) + \sum_{r \in \mathcal{R}} y_r \lambda_{s(r)} - \sum_{b \in \mathcal{B}} C_b(\sum_{r:b \in r} y_r) \end{aligned}$$

where $\lambda = (\lambda_s, s \in \mathcal{S})$, is a vector of Langrange multipliers. Then

$$\begin{aligned}\frac{\partial L}{\partial x_s} &= U_s^{'}(x_s) - \lambda_s \\ \frac{\partial L}{\partial y_r} &= \lambda_{s(r)} - \sum_{b\in\mathcal{B}} C_b^{'}(\sum_{r:b\in r} y_r) = \lambda_{s(r)} - \sum_{b\in\mathcal{B}} \mu_b\end{aligned}$$

where μ_b is interpreted as the *shadow price* [41] of block *b*; the shadow price can be interpreted as the price per unit bandwidth at block b,

$$\mu_b = p_b(\sum_r A_{br} y_r) \text{ with } p_b(y) = \frac{dC_b(y)}{dy}$$

Hence, at a maximum of L[1] over the orthant $x, y \geq 0$, the following conditions hold:

$$\begin{aligned}\lambda_s &= U_s^{'}(x_s) \quad \text{if } x_s > 0 \\ &\geq U_s^{'}(x_s) \quad \text{if } x_s = 0 \qquad \text{(A.5)} \\ \lambda_{s(r)} &= \textstyle\sum_{b\in r} \mu_b \quad \text{if } y_r > 0 \\ &\leq \textstyle\sum_{b\in r} \mu_b \quad \text{if } y_r = 0 \qquad \text{(A.6)}\end{aligned}$$

Formula (A.6) implies that if route r has positive flow on it, $y_r > 0$, then necessarily $\sum_{b\in r} \mu_b \leq \sum_{b\in r^*} \mu_b$ for any other route r^* that serves the same source-destination pair. For one route r, the sum $\sum_{b\in r} \mu_b$ is called the route shadow price.

If we view a terminode network as being composed of a number of blocks, we can expect that will be many available routes from the source block to the destination block. The previous derivations show that there are source rates and routes towards destinations such that the optimum of the global utility function is achieved. It is shown that sources should be rate adaptive, and they should distribute their flow over the routes that have the smallest route shadow price. Denote by $p^*(s)$ minimum route shadow price for source s. Then from above, source should set its rate to $x_s^* = U_s^{-1}(p^*(s))$. Source can then use multiple paths to send the rate (x^*). Then, the source should split this rate *arbitrarily* among paths that have the same route shadow price (equal to $p^*(s)$), such that the sum of rates on this paths equals to x_s^*.

We presented the solution of the optimal routing problem. However, here we do not present

[1] since U is concave and C are convex then the conditions given in the upper section are sufficient conditions in order to optimize global utility. There is a unique vector y for the global maximum

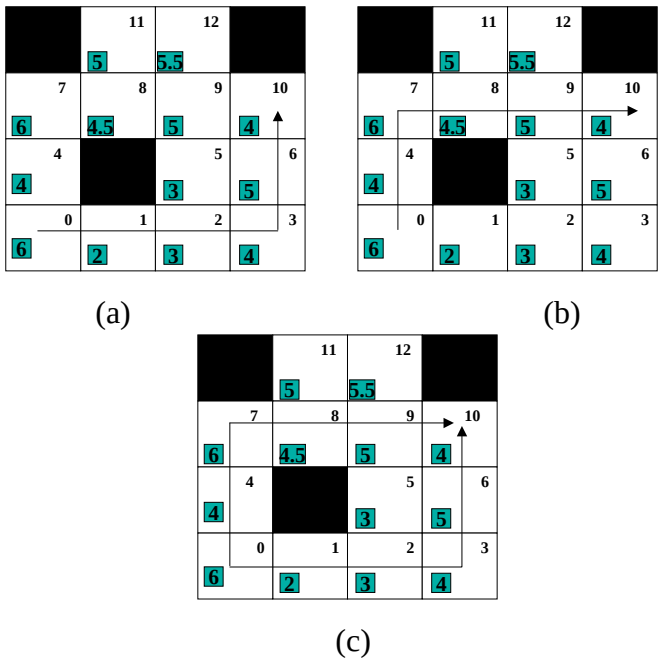


Figure A.2: Figure presents the optimal path from the source in block 0 to the destination in block 10 for different background flows. (a) Background flows are: 3->9, 2->11, 5->12, 4->2, 7->3, 4->12, 0->11. (b) remove flow 0->11 and add two flows. optimal path changes. (c) remove the last two flows, the optimal route is now split in two routes.

the distributed algorithm that every node locally runs in order to solve a global optimization problem. Some ideas of how to design such a distributed algorithm can be found in [45, 46, 47].

A.3 An example of optimal routing

One example of the optimal routing is presented in Figure A.2. A terminode network is composed of 16 blocks. Three blocks, colored in black, cannot be used for packet forwarding. For the other blocks, the maximal capacity of each block is presented with the number in the low left corner.

We look at the optimal routing from the source in block 0 to the destination in block 10 for different background flows. The optimal routing problem can be expressed as follows: maximize the global utility as defined with Equation A.1 subject to flow conservation constraints and capacity constraints for each block. The flow conservation constraints mean that for a given block, and a given flow, the amount of flow that comes into and goes out the block has to be the same. The flow conservation constraints ensure the desired traffic is routed from the source to the destination block. For utility and cost functions, we use piece-wise linear functions, as presented in

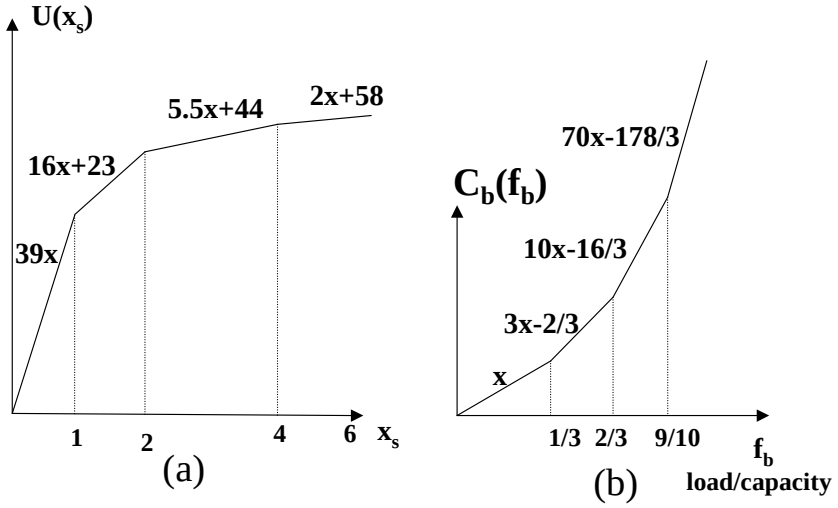


Figure A.3: (a)- concave utility function $U(x_s)$, x_s is transmission rate of source, (b)- convex cost function $Cb(f_b)$, $f_b = l_b/c_b$ where l_b is load in block b, c_b is capacity of block b

Figure A.3. When utility and cost functions are piece-wise linear, optimal routing is a complete linear programming problem.

Formulation of the optimal routing problem as the linear program

We model a terminode network as being composed of a set of blocks $\mathcal{B}$. A block b is associated with capacity c_b. Denote by $\mathcal{N}$ a set of pairs of neighbouring blocks. $(b_x, b_y) \in \mathcal{N}$, if the flow from b_x can go to b_y, and vice versa. For example, in Figure A.2, neighbouring blocks of block 4 are blocks 0 and 7.

We have set $\mathcal{S}$ of source-destination pairs. For a given block b_y and its neighbouring block b_x, and a given source-destination pair $(s,d) \in \mathcal{S}$ we associate a variable $f^{(s,d)}_{(b_x,b_y)}$ telling how much of the traffic flow (s,d) goes from b_x to b_y. U_s is used to model the piece-wise linear utility function of source s. C_b models the piece-wise linear cost function of block b. For each block b, we associate load l_b, i.e. the sum of the flows going over block b.

The optimal routing problem can be defined as the following linear programming problem:

$$\max U(x) = \sum_{s \in \mathcal{S}} U_s(x_s) - \sum_{b \in \mathcal{B}} C_b(l_b), \tag{A.7}$$

subject to

$$\sum_{b_x:(b_x,b_y)\in\mathcal{N}} f^{(s,d)}_{(b_x,b_y)} - \sum_{b_z:(b_y,b_z)\in\mathcal{N}} f^{(s,d)}_{(b_y,b_z)} \quad = -x_s \quad \text{if } s \in b_y,$$
$$= x_s \quad \text{if } t \in b_y,$$
$$= 0 \quad \text{otherwise},$$
$$b_y \in \mathcal{B}, (s,d) \in \mathcal{S}, \tag{A.8}$$

The load of block $b_y \in \mathcal{B}$ is given by the following formula,

$$l_{b_y} = \sum_{b_x:(b_x,b_y)\in\mathcal{N},(s,d)\in\mathcal{S}} f^{(s,d)}_{(b_x,b_y)} \tag{A.9}$$

Given that the utility and cost functions are piece-wise linear and presented in Figure A.3, we have the following constraints:

$$U(x_s) \leq 39x_s, \quad (s,d) \in \mathcal{S} \tag{A.10}$$
$$U(x_s) \leq 16x_s + 23, \quad (s,d) \in \mathcal{S} \tag{A.11}$$
$$U(x_s) \leq 5.5x_s + 44, \quad (s,d) \in \mathcal{S} \tag{A.12}$$
$$U(x_s) \leq 2x_s + 58, \quad (s,d) \in \mathcal{S} \tag{A.13}$$
$$C_b \geq l(b)/c_b, \quad b \in \mathcal{B} \tag{A.14}$$
$$C_b \geq 3l_b/c_b - 2/3, \quad b \in \mathcal{B} \tag{A.15}$$
$$C_b \geq 10l_b/c_b - 16/3, \quad b \in \mathcal{B} \tag{A.16}$$
$$C_b \geq 70l_b/c_b - 178/3, \quad b \in \mathcal{B} \tag{A.17}$$

Constraints (A.8) are flow conservation constraints; constraints (A.9) define the load in each block and constraints (A.10) to (A.13) define the utility of sources, and constraints (A.16) to (A.17) define the cost in each block.

We used Mathematica to solve the described linear programming problem. The solution to the optimal routing problem gives optimal routes and rates on all routes for all source-destination pairs. As a result of the optimization, optimal routes in the network are found. These routes are presented in Figure A.2. This figure illustrates that optimal routes are subject to the number of source-destination pairs. The example in A.2 (c) shows that the optimal route is given by two routes.

Appendix B

Perimeter-mode packet forwarding - pseudocode

In this appendix we present the perimeter-mode packet forwarding pseudocode. As it is described in Section 4.6, where the operation of GPF is described, the packet forwarding switches from the greedy-mode to the perimeter-mode whenever the greedy-mode packet forwarding fails. This happens when the current node to forward the packet does not have a neighbour that is closer to the referent location (LOC). LOC can be either an anchor, or the destination location.

We have implemented perimeter-mode of packet forwarding in GloMoSim [74], as a part of terminode routing (described in Chapters 4 and 5). We used the same perimeter-mode algorithm as in [38], and the pseudocode is taken from [38]. To ensure the completeness of the perimeter-mode forwarding description, its building blocks are presented in pseudocode (Figures B.2, B.3, B.4, B.5). Figure B.1 illustrates the operation of planar graph face traversal, and Table B.1 presents the packet header fields necessary for perimeter-mode forwarding.

Field	Function
LOC	reference location (anchor or LDA_D)
L_p	LDA of node where the packet entered perimeter mode
L_f	point on line L_p-LOC packet entered current face
e_0	first edge traversed on current face
$mode$	packet mode: greedy or perimeter
n_{in}	the previous hop LDA

Table B.1: Packet header fields used in perimeter-mode forwarding

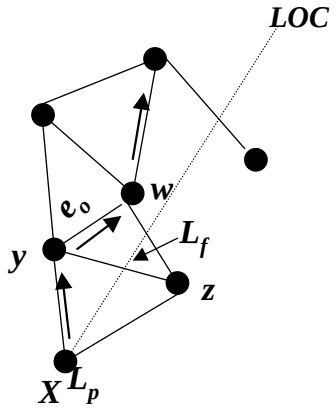


Figure B.1: The planar graph traversal method: LOC is the destination; X is the node where the packet enters perimeter mode. Solid arrows are forwarding hops.

```
GPF_perimeter(p,LOC):
    switch
      case p.mode==greedy:
         p.mode = perimeter
             p.L_p=p.L_f=self.LDA, p.n_in=self
         t=PERI-INIT-FORWARD(p)
         p.e_0=(self.EUI,t)
         transmit(p,t)
      case p.mode==perimeter:
         if DISTANCE(self.LDA,LOC)<DISTANCE(p.L_p,LOC)
            then p.mode=greedy; greedy_forward(p)
         else t =RIGHT-HAND-FORWARD(p,p.n_in)
            if p.e_0==(self.EUI,t)
               then drop p //destination unreachable
            else t=FACE-CHANGE(p,t)
         p.n_in=self, transmit(p, t)
```

Figure B.2: GFP perimeter-mode packet forwarding. The reference location is LOC

```
RIGHT-HAND-FORWARD(p, n_in):
    b_in=NORM(ATAN2(self.LDA.y-n_in.y,self.LDA.x-n_in.x))
    δ_min = 3π
    for each (a, l) in N // N is a list of neighbours
    do if a==n_in
       then continue
//NORM normalizes its agrument in radian into [0,2π] by repeatedly adding 2π.
ATAN(y, x) computes the arc tangent of y/x
       b_a=NORM(ATAN2(self.LDA.y-l.y,self.LDA.x-l.x))
       δ_b=NORM(b_a-b_in)
       if δ_b < δ_min
            δ_min=δ_b; a_min=a
return a_min
```

Figure B.3: The RIGHT-HAND-FORWARD algorithm. The previous hop is n_{in}

```
FACE-CHANGE(p, t):
    i=INTERSECT(t.l,self.LDA, p.L_p, LOC)
    if i ≠ NIL
       then if DISTANCE(i, LOC)<DISTANCE(p.L_f, LOC)
            then p.L_f = i
                 t =RIGHT-HAND-FORWARD(p, t)
                 t =FACE-CHANGE(p, t)
                 p.e_0 =(self.EUI,t)
return t
```

Figure B.4: The FACE-CHANGE algorithm

```
PERIM-INIT-FORWARD(p):
    b_in=NORM(ATAN2(self.LDA.y-p.LOC.y,self.LDA.x-p.LOC.x))
    δ_min = 3π
    for each (a, l) in N
    do b_a=NORM(ATAN2(self.LDA.y-l.y,self.LDA.x-l.x))
       δ_b=NORM(b_a-b_in)
       if δ_b < δ_min
            δ_min=δ_b; a_min=a
return a_min
```

Figure B.5: Initialization of the perimeter-mode forwarding

Appendix C

Simulation Parameters for AODV and LAR1

Tables C.1 presents the simulation parameters that are used in AODV implementation in GloMoSim [74].

	Parameter	Value
General	net_diameter	35
	node_traversal_time	40ms
	active_route_to	10s
	rrep_wait_time	2.1s
	bcast_id_save	30s
Expanding Ring Search	ttl_start	1
	ttl_increment	2
	ttl_threshold	7

Table C.1: AODV parameters

The net_diameter presents the approximate diameter of the network, and is used for setting the TTL value of broadcast control packets. The node_traversal_time represents an estimation of the processing time of a packet at a node. It is used to estimate how long a node should wait to receive a route reply after broadcasting a route request.

The expanded ring search is used to enhance AODV. For the expanded ring search, the initial TTL of the route request packet (ttl_start) is set to one. Each time a reply is not received within the time specified with the rrep_wait_time parameter, the TTL is incremented by ttl_increment, until the threshold (ttl_threshold) is reached. After that, the source broadcasts the route request

packet across the network. If the route is not used for the interval equal to the active_route_to parameter, it is erased from the route cache. The bcast_id_save parameter defines the time a node keeps in its cache an already seen route request packet(in order to avoid redundant transmissions of route request control packets).

Table C.2 presents the simulation parameters that are used in LAR1 implementation in GloMoSim. No DSR optimization features are included in LAR1.

Parameter	Value
lar1_request_seen_lifetime	30s
lar1_req_timeout	2s

Table C.2: LAR1 parameters

LAR [92] specifies that in order to avoid redundant transmissions of route request control packets, intermediate nodes memorize the seen route request control packets. The lar1_request_seen_lifetime parameter is the time a node keeps in its cache an already seen route request packet. lar1_req_timeout specifies the time the source wait for the route reply control packet from the destination. If the route reply is not received, the source then re-initiate route discovery. Then a route request is retransmited via pure flooding.

Appendix D

Curriculum Vitae and List of Publications

Ljubica P. Blažević was born in Loznica, Yugoslavia. She received the BSc degree in electrical engineering in 1993 from the Faculty of Electrical Engineering, Belgrade, Yugoslavia.

From 1993-1996 she worked as a R&D engineeer at the Institute “Mihajlo Pupin” in Belgrade.

From 1996-1997 she attended the Doctoral School in Communication Systems at EPFL, Switzerland.

In 1997 she joined the Institute for Computer Communications and Applications (ICA) at the Department of Communication Systems, EPFL. From 1997-1999 she participated in the EPFL research project “Multicast Routing and Mobility” where she was responsible for the design of the multicast routing protocol to support Internet host mobility. From 1999-2002 she participated in the EPFL research project “Towards Mobile Ad Hoc WANs: Terminodes”, where she worked on the solution for packet forwarding in an autonomous, fully self-organized wireless network (mobile Ad Hoc Network). Her current research interests include routing problems in mobile systems and networked group communications in the Internet.

Publications

- L. Blazevic, S. Giordano, J.-Y. Le Boudec, “Self Organized Terminode Routing”, *Cluster Computing Journal*, Vol.5, No.2, April 2002.

- L. Blazevic, L. Buttyan, S. Capkun, S. Giordano, J.-P. Hubaux, J.-Y. Le Boudec, “Self-Organization in Mobile Ad Hoc Networks: the Approach of Terminodes”, *IEEE Communications Magazine*, June 2001, Vol.39, No.6.

- L. Blazevic, J.-Y. Le Boudec, "Distributed Core Multicast (DCM): a multicast routing protocol for many groups with few receivers", *ACM SIGCOMM Computer Communication Review*, October 1999, Vol.29 No.5.

- L. Blazevic, J.-Y. Le Boudec, S. Giordano, "A Scalable Routing Scheme for Self-Organized Terminode Network", *Communication Networks and Distributed systems modeling and Simulation conference (CNDS)*, San Antonio, Texas, January 2002.

- L. Blazevic, S. Giordano, J.-Y. Le Boudec, "Self Organized Routing in Wide Area Mobile Ad Hoc Network", *IEEE Symposium on Ad Hoc Wireless Networks (Globecom 2001)*, San Antonio, Texas, November 2001.

- L. Blazevic, S. Giordano, J.-Y. Le Boudec, "Self Organized Terminode Routing Simulation", *Proceedings of ACM International Workshop on Modeling Analysis and Simulation of Wireless and Mobile systems (MSWiM 2001)*, Rome, Italy, July 2001.

- L. Blazevic, S. Giordano, J.-Y. Le Boudec, "Self-Organizing Wide-Area Routing", *Proceedings of SCI 2000/ISAS 2000*, Orlando, July 2000.

- S. Giordano, M. Hamdi, J.-P. Hubaux, J.-Y. Le Boudec, L. Blazevic, "Issues on Mobile Ad Hoc WANs", *IEEE ICME 2000*, NY - US, July 2000.

- L. Blazevic, J.-Y. Le Boudec, "Distributed Core Multicast (DCM): a routing protocol for many groups with few receivers", *Proceedings of First International Workshop on Networked Group Communication*, Pisa, November 1999, pp. 108-125.

- L. Blazevic, J.-Y. Le Boudec, "Scalable IP multicast for many very small groups with many senders and its application to mobility", *Proceedings of 3rd European Personal Mobile Communications Conference (EPMCC'99)*, Paris, France, March 1999.

- L. Blazevic, E. Gauthier, "A Scalable Protocol For Reporting Periodically Using Multicast IP", *Proceedings of IFIP TC-6 Eighth International Conference on High Performance Networking (HPN'98)*, Vienna, Austria, September 1998

- M. Stojanovic, S. Nedic, L. Blazevic, "A Multipoint Control Unit For Data Conferencing of Low Bit Rate Multimedia Terminals", *Proceedings of the Seventh International Conference on Signal Processing Applications and Technology*, Boston, MA, USA, October, pp.1302-1306, 1996.

- M. Stojanovic, L. Blazevic, S. Nedic, “A Concept of Multipoint Control Unit For Data Conferencing of Low Bit Rate Multimedia Terminals”, *Proceedings of the YUINFO’96*, Brezovica, Yugoslavia, April, 1996.
- L. Blazevic, M. Stojanovic, “Implementation of communication services used by a MHS application”, *Proceedings of the ETRAN*, Zlatibor, Yugoslavia, June, 1995.
- L. Blazevic, J. Radunovic. “Simulation of the time response of the photodetector ”, *Proceedings of the MIOPEL’93*, Nis, Yugoslavia, October 1993.
- L. Blazevic, J. Y. Le Boudec, “Distributed Core Multicast (DCM): a routing protocol for many small groups with application to mobile IP telephony”, *Internet draft* (draft-blazevic-dcm-mobility-00.txt), June 2000.