INSTITUTO SUPERIOR TÉCNICO
Universidade Técnica de Lisboa

Heterogeneity-Aware Multicast for MANETs

Oksana Denysyuk

Dissertação para obtenção do Grau de Mestre em
Engenharia Informática e de Computadores

Júri

Presidente:	Professor Joaquim Armando Pires Jorge
Orientador:	Professor Luís Eduardo Teixeira Rodrigues
Vogais:	Professor António Manuel Raminhos Cordeiro Grilo

October 2009

Acknowledgments

First of all, I would like to express my profound gratitude to my advisor Professor Luís Rodrigues. I consider a privilege to have an opportunity to work with him. His persistence and the highest level professionalism made the greatest contribution to this work.

I also thank to all my colleagues from the GSD group at INESC-ID for all our fruitful discussions and their seamless help in keeping me motivated during this hardworking year.

I also thank to all my friends that have always been around when I most needed. Their cheerfulness has always helped me keep going.

And at last but not least, my eternal and heart-felt gratitude goes to my family. My parents' unconditional love and support is the most precious treasure in my life. I am also deeply grateful to my soul mate for his love and priceless emotional endorsement along all these years. And finally, I would like to express a very special gratitude to my darling twin sister that has always stayed by my side with her mirror-like nature, pretty much alike but always the opposite.

Lisboa, October 2009
Oksana Denysyuk

To My Parents

Resumo

A difusão em grupo é útil para suportar diferentes aplicações em redes móveis e ad-hoc, incluindo a disseminação de dados, descoberta de serviços, publicação-subscrição, entre outras. Consequentemente, este problema tem sido extensivamente estudado. No entanto, a maior parte dos protocolos de difusão são desenhados para um padrão específico de mobilidade e, portanto, são incapazes de atingir um bom desempenho perante condições de mobilidade heterogénea.

Esta dissertação propõe e avalia o HAMP, um protocolo de difusão para redes móveis que combina a eficiência de abordagens estruturadas e a robustez de esquemas baseadas em inundação, adaptando o seu algoritmo de encaminhamento de acordo com as condições de mobilidade observadas localmente por cada nó. Se muita mobilidade é observada na vizinhança, os nós passam a encaminhar as mensagens em inundação limitada. Assim que a rede se estabiliza os nós revertem para o encaminhamento estruturado.

O HAMP foi concretizado como uma combinação do protocolo em malha PUMA e do PAMPA como uma forma de inundação localizada para atravessar zonas de instabilidade. Contudo, as ideias-chave da solução podem ser aplicadas a outros algoritmos de encaminhamento.

Resultados experimentais obtidos com o simulador de rede NS-2 mostram que o HAMP consegue combinar de forma eficaz a robustez de inundação com a eficiência de encaminhamento em malha.

Abstract

Multicast is an essential group communication service and an important building block for many applications in MANETs, including data dissemination, service discovery, publish-subscribe, among others. Therefore, it has been widely studied and many solutions can be found in the literature. However, most existing multicast protocols are tailored to a specific type of mobility pattern and therefore are unable to excel in face of heterogeneous mobility conditions.

This thesis proposes and evaluates HAMP, a Heterogeneity-Aware Multicast Protocol for MANETs that combines the efficiency of structured approaches and the robustness of flooding-based schemes. HAMP dynamically adapts its forwarding mechanism according to locally observed mobility conditions. If much mobility is observed in the neighborhood the node starts forwarding multicast traffic using a scoped flooding technique. As local network conditions stabilize the node reverts to a structured mesh-based operation.

HAMP was implemented as a combination of a mesh-based protocol that uses a flooding-based technique. However, the main idea of combining structured and unstructured approaches in the same protocol can be also applied to other forwarding schemes.

Experimental results obtained using the NS-2 network simulator confirm that HAMP successfully combines the robustness of flooding with the efficiency of mesh-based routing.

Palavras Chave
Keywords

Palavras Chave

Redes Móveis e Ad-Hoc

Comunicação em Grupo

Difusão em Grupo

Protocolo de Encaminhamento.

Keywords

MANETs

Group Communication

Multicast

Routing Protocol.

Contents

List of Figures

List of Tables

Acronyms

ADB Adaptive Backbone-Based Multicast

ADMR Adaptive Demand-Driven Multicast Routing Protocol

AMRIS Ad hoc Multicast Routing protocol utilizing Increasing id-numbers

AMRoute Ad hoc Multicast Routing Protocol

CBT Core-Based Tree

DCMP Dynamic Core Multicast Routing Protocol

DDM Differential Destination Multicast

DFN Downstream floodgate node

FH Flooding horizon

FN Floodgate node

GPS Global Positioning System

HAMP Heterogeneity-Aware Multicast Protocol

MACT Multicast Activation message

MANET Mobile Ad Hoc Network

MAODV Multicast On Demand Distance Vector Routing Protocol

NSRMP Network Sender Multicast Routing Protocol

ODMRP On Demand Multicast Routing Protocol

PAMPA Power-Aware Message Propagation Algorithm

PUMA Protocol for Unified Multicasting through Announcements

RREP Route Reply message

RREQ Route Request message

TTL	Time to live
UFN	Upstream floodgate node

1

Introduction

A *MANET* (Mobile Ad Hoc Network) is formed by a set of mobile wireless devices with no fixed topology. The nodes can move freely, leave and enter the network at any time. Typically, nodes communicate in a peer-to-peer fashion by using the wireless radio medium. In a MANET, there is no distinction between a host and a router, since all nodes can be sources as well as traffic forwarders. The ease of deployment makes MANETs attractive to variety of application areas, such as disaster recovery operations, search and rescue, military operations, ad hoc gaming, etc. Many of these applications benefit from services such as group oriented computing, multimedia streaming, video conferencing, and interactive information sharing. In turn, all these services may benefit from the availability of multicast support. Multicast, the ability to send a message to a group of processes, is therefore a central component of any group communication service. This dissertation focuses on multicast protocols for mobile ad hoc environments.

Multicast is an important building block for many applications in MANETs, including data dissemination (Drabkin, Friedman, Kliot, & Segal 2007), publish-subscribe (Voulgaris, Riviere, Kermarrec, & van Steen 2006), among others. Therefore, a significant amount of work has been performed in designing multicast protocols for the MANET environment. MANETs have a set of properties that distinguish them from the wired environment. Nodes in a MANET are resource-constrained, with scarce processing, storage, and battery resources. Moreover, the absence of a fixed infrastructure and the mobility of nodes make the network subject to frequent disconnections and topology changes. Multicast protocols attempt to minimize the overhead of the protocol, namely in terms of control data that needs to be exchanged to support multicast. The challenge is to achieve a high reliability level, in face of node mobility and topology changes, without sacrificing efficiency.

1.1 Motivation

Multicast protocols for MANETs can be divided into two main classes: unstructured and structured approaches. Unstructured approaches are based on some form of optimized flooding. They are oblivious (or adapt well) to topology changes and, therefore, are better suited for scenarios with fast mobility patterns, where it is hard to maintain stable routes among nodes. Structured approaches create some form of multicast-tree. They trade the cost of building and maintaining the structure for a more efficient message dissemination procedure in stable conditions. They are therefore more suitable to scenarios with low or sporadic mobility.

This thesis is motivated by the belief that many MANET deployments of the future will not be homogeneous in terms of mobility patterns. For instance, MANETs created for disaster management will have a mix of quasi-stable nodes (command center, field-hospital) and highly mobile nodes (search and rescue teams); conventions have a mix of fixed nodes (stands) and mobile nodes (attendees); airports and universities have people in transit but also people waiting in coffee-shops, restaurants or reading rooms. To the best of our knowledge, there is no multicast protocol that excels in such heterogeneous environments.

1.2 Contributions

The thesis addresses the problem of supporting multicast in MANETs with heterogeneous mobility, namely in MANETs where high-mobility and low-mobility regions co-exist. In particular, the thesis makes the following contributions:

- It proposes a novel approach to combine structured and unstructured multicast protocols, that relies on the identification of special nodes, called *floodgate nodes*, that commute between the two message forwarding modes.
- It illustrates the use of the technique, describing *HAMP – Heterogeneity-Aware Multicast Protocol*, an efficient and yet robust multicast protocol that combines *PUMA - Protocol for Unified Multicasting through Announcements*, a structured mesh-based multicast algorithm for MANETs, and *PAMPA - Power-Aware Message Propagation Algorithm*, a flooding technique.

1.3 Results

The results produced by this thesis can be enumerated as follows:

- A specification of the HAMP multicast protocol and its implementation for the NS-2 simulation platform.
- An extensive evaluation of HAMP comparing it with PUMA, a structured mesh-based multicast algorithm for MANETs, and PAMPA, a flooding technique.

Based on experimental results, we show that HAMP offers, under heterogeneous mobility conditions, a very favorable tradeoff between the efficiency of structured approaches and robustness of flooding mechanisms.

1.4 Research History

This work was performed in the context of the FCT PTDC/EIA/71752/2006 “Redico: Reconfiguração Dinâmica de Protocolos de Comunicação” project. One of the goals of the project is to build group communication protocols that are able to adapt to heterogeneous operational conditions. During my work, I benefited from the fruitful collaboration of the remaining members of the Redico and “Grupo de Sistemas Distribuídos” at INESC-ID team members, namely José Mocito, Liliana Rosa and João Leitão.

The text of this dissertation includes material that has been previously published in (Denysyuk, Mocito, & Rodrigues 2009).

1.5 Organization of the Dissertation

In Chapter 2, we survey the existing message dissemination techniques used to implement multicast in mobile environments. We present both structured and unstructured approaches and discuss the benefits and drawbacks of each solution. In Chapter 3, we propose and describe HAMP. Chapter 4 presents the evaluation results and discusses the proposed protocol making a comparison with other structured and unstructured approaches. Finally, Chapter 5 presents the conclusions of this dissertation and sketches an outline of the future work.

2

Related Work

A multicast service allows processes to send a message to a *group* of recipients. Typically, group membership is dynamic, i.e., a node in the MANET may leave or join the group at any moment. Furthermore, to each group is associated a *multicast address* which can be used to identify the group of recipients. Therefore, the sender is not required to name explicitly the identities of all group members; it is up to the multicast service to ensure that the message is delivered to all group members in the most efficient manner.

The multicast service can be offered with different levels of reliability and ordering guarantees, usually by stacking different protocol layers. The fundamental layer for any practical multicast service is a best-effort multicast primitive. In stable and failure-free runs, the best-effort service delivers the multicast message to all group members. However, omissions at the data link, failures, collisions or node movement may cause a multicast message to be delivered to just a subset of the members. Thus the best-effort multicast service can be complemented with protocols that are able to detect and recover from such faults.

In this Chapter, we survey different techniques to implement multicast in MANETs. We will be mainly concerned with protocols that provide a best-effort service.

2.1 Broadcast in MANETs

Broadcast can be seen as a particular case of multicast, where all nodes are intended to receive the message. Broadcast is one of the most fundamental services in a MANET since it is used as a building block for many services and applications, such as (unicast and multicast) routing protocols, service discovery and information dissemination among others. The most straightforward way to implement broadcast in mobile environments is by using flooding, as described below.

2.1.1 Flooding

Flooding consists in having each node rebroadcasting a message to its neighbors upon receiving it for the first time. More precisely, flooding can be implemented as follows: a source node broadcasts a message to all its neighbors. Each node checks if it has received the message for the first time, in which case the node rebroadcasts the packet (nodes are required to keep the identity of previously flooded messages for some amount of time). The procedure is repeated at every node until all the members of the network have received the message.

Flooding usually covers the entire network, but can also be limited by TTL (time to live) parameter. In this case, a node receiving the flooded message only rebroadcasts it if the message's TTL is greater than 0. The TTL is decremented in every retransmission.

The algorithm described above, also called *simple flooding* (Obraczka, Viswanath, & Tsudik 2001), has the main advantage of being a very straightforward approach: it requires little memory and computation resources from the network nodes. But, unfortunately, simple flooding is usually very costly in terms of communication overhead, and will result in serious redundancy, contention and collision, a phenomenon also known as *broadcast storm* (Ni, Tseng, Chen, & Sheu 1999). To address this problem, many alternatives to simple flooding have been proposed in the literature.

Optimized alternatives to simple flooding can be categorized in following classes: *probability based methods* where nodes decide to rebroadcast based on some probability function; *counter based methods* based on number of retransmissions of the previously seen packet; *area based methods*, where nodes decide to rebroadcast based on an estimate of the number or location of their neighbors; and *neighbor knowledge methods* where nodes decide to rebroadcast based on information they receive from their neighbors. The main goal of all these methods is to reduce the number of redundant transmissions; this is achieved at the cost of some additional algorithm complexity and extra computing and memory resources.

2.1.1.1 Probability Based Methods

As the name implies, in probability based methods, every node retransmits a message with some predetermined probability p. An example of probability based flooding is GOSSIP1 (Haas, Halpern, & Li 2006). The source of a message sends it with probability 1. When a node first receives a message, with probability p it broadcasts it to the neighbors and with probability $1 - p$ it discards the message. The parameter p has a fixed value defined at deployment time. The main problem of this approach is that, if a source has few neighbors, there is a chance that none of the neighbors rebroadcasts the message and the message is not propagated further.

2.1.1.2 Counter Based Methods

The *counter based scheme* operates by estimating the node density dynamically. Upon reception of a previously unseen message, the node initiates a counter and sets a timer to a randomly chosen value. The counter is incremented for each redundant message received. If the timer expires before a predefined *counter threshold* has been reached, the message is retransmitted. Otherwise the message is dropped. This technique is based on the inverse relation between the number of redundant messages received and the *expected additional coverage*, the area covered by the node excluding the already covered by other hosts (Ni, Tseng, Chen, & Sheu 1999).

2.1.1.3 Area Based Methods

Area based methods usually rely on the notion of *distance* or *location* to decide if a node should retransmit a broadcast packet.

- *Distance based schemes* assume that every node is able to determine its relative distance to the neighbors. This can be done by using the signal strength of the received message. If the distance to the sender is very short, the expected additional coverage is minimal or null, thus, the message is not retransmitted. PAMPA (Miranda, Leggio, Rodrigues, & Raatikainen 2006) is an example of a distance based broadcast algorithm that uses the receiving power to estimate the distance to the source; it sorts the receiving nodes using the estimated distance to the source such that nodes more distant to the source are more likely to retransmit first. In practice, nodes delay the retransmission by an amount of time that is proportional to the measured signal strength. The rebroadcasting is canceled if, during the *delay* period, a retransmission of the same message is heard. That will prevent nodes providing a small additional coverage from retransmitting.

- *Location based schemes* require information about the location of nodes in the physical space. Such methods may be supported by some positioning service such as GPS (Global Positioning System). Every node piggybacks its location information to the original broadcast message. This information can be used to estimate the expected additional coverage. An example of a location based algorithm is Six-Shot Broadcast (Garbinato, Holzer, & Vessaz 2008). The protocol assumes the existence of GPS location service. When a node wishes to rebroadcast a message, it uses the location service to choose 6 neighbors, which are closer to the vertices of a hexagon centered at the source, to propagate a message in all geographical directions.

2.1.1.4 Neighbor Knowledge Methods

Neighbor knowledge methods rely on the explicit exchange of information among physical neighbors. In this case, nodes are required to periodically send beacon messages that allow each node to become aware of its one-hop (and, in some cases, two-hop) neighbors. Depending on the protocol, nodes may also be required to gather the state of these neighbors (for instance, available battery).

An example of a neighbor knowledge protocol is RAPID (Drabkin, Friedman, Kliot, & Segal 2007) that calculates its broadcast probability according to the number of the node's one-hop neighbors. After receiving the packet for the first time, the node waits a small random period before rebroadcasting. If, during this time, the node does not hear any other retransmission of the same packet, it applies a probability function to decide if it will retransmit the packet. The probability function depends on the number of neighbors and a reliability factor β related to the number of nodes that should perform a retransmission in a one hop neighborhood.

A node that decides not to retransmit continues to monitor the network for an additional random period of time. This second monitoring period has a larger interval. The node will retransmit with probability 1 if it does not hear at least one retransmission of the message during this period.

In order to increase reliability of data dissemination, RAPID also employs a recovery technique that operates as follows. Every node periodically broadcasts the headers of the messages it received from other nodes, a procedure named *gossiping*. If a node receives the header of a missing message, it requests the original message from the gossiping node. In this way, if a node fails to receive some messages during the flooding phase, it may still recover it later.

2.2 Multicast in MANETs

Multicast differs from broadcast since it aims at delivering a message to only a (typically small) subset of the entire set of nodes. Obviously, multicast may be implemented by using flooding: all nodes would participate in the message dissemination but only the interested nodes would deliver the message to the upper layers. As it will be discussed further, this can even be the most appropriate strategy in a highly dynamic network, where due to the nodes mobility the topology is constantly changing and routing information quickly becomes stale. However, using flooding to implement multicast in stable networks is clearly a sub-optimal approach. The goal of a multicast protocol is to limit the number of nodes involved in the multicast operation while still delivering the message to all group members.

Multicast protocols can be distinguished according to different aspects, including: at which level of the protocol stack they are implemented; if routes are created proactively or reactively; and what kind of structure is maintained to support multicast.

Considering the level of the protocol stack in which multicast is implemented we can distinguish two main alternatives.

- One approach consists in implementing multicast on top of an existing unicast routing protocol (without changing the unicast protocol). This approach is known as *application level multicast* or *overlay multicast*; it operates by having group members coordinate in order to deliver the multicast message to the group, by exchanging the messages among them using the underlying unicast primitive. In an overlay approach, only the group members have to maintain additional information about the group. This method also provides more interoperability with an existing infrastructure as only the group members have to run the multicast protocol.

- The other alternative consists in implementing multicast at the *network layer*, possibly by augmenting a unicast routing protocol with multicast support. By using this method communication cost and message delivery delays may be reduced as instead of sending via multiple unicasts, data is sent to all the recipients at the same time. Thus, network layer multicast protocols perform better in minimizing resource consumption and data delivery latency.

Considering when routes are created multicast routing algorithms may be divided in two major groups: *proactive* and *reactive*.

- Proactive schemes maintain routing information among all nodes in the network all the time. As route information is always available and up-to-date, proactive protocols usually deliver messages with lower latency. That is achieved at the cost of constant network overhead. *CBT - Core-Based Tree* (Ballardie, Francis, & Crowcroft 1993) is an example of a proactive routing protocol.
- Reactive schemes only construct the route to the multicast receivers when necessary. They normally induce smaller signaling overhead but can suffer from larger packet delivery delays due to their on-demand nature. MAODV (Royer & Perkins 1999), ODMRP (Lee, Su, & Gerla 1999), DCMP (Das, Manoj, & Murthy 2002), NSMRP (Farhan 2008) are only a few examples of reactive multicast protocols.

Finally, considering the type of topology created by the routing protocol, multicast protocols are often categorized in the four following groups:

- *Tree-based* approaches create non-redundant routes between the members. The way such structures are constructed tends to make them match the underlying physical topology, making data dissemination very efficient. On the other hand, tree topologies are very sensitive to failures, mobility and partitioning: as soon as a tree member leaves or crashes, the tree breaks and data dissemination becomes compromised until the tree is healed. MAODV (Royer & Perkins 1999) and AMRIS (Wu & Tay 1999) are examples of tree-based protocols.
- *Mesh-based* approaches allow multiple routes from senders to receivers. This approach has some advantages over the tree-based structures. Namely, a mesh tolerates better node failures and mobility. In addition, the existence of multiple paths in a mesh may be used to adapt the routes, for example, for load balancing or partition recovery. ODMRP (Lee, Su, & Gerla 1999) and DCMP (Das, Manoj, & Murthy 2002) are examples of such protocols.

- *Stateless* multicast does not require any additional information be maintained by the nodes. This approach assumes the existence of an underlying unicast protocol. The node wishing to send a message to a multicast group explicitly enumerates all the multicast receivers. Stateless multicast is suitable for small multicast groups. DDM (Ji & Corson 2001) is an example of a stateless multicast protocol.

- *Hybrid* approaches combine some of the above techniques. Various protocols first build a mesh-based topology and then derive a data dissemination tree on top of the mesh. AMRoute (Xie, Talpade, Mcauley, & Liu 2002) is an example of such a protocol.

In the next paragraphs we briefly describe some relevant protocols that illustrate the several design choices above.

2.2.1 MAODV

The *MAODV - Multicast On Demand Distance Vector Routing Protocol* (Royer & Perkins 1999) is one of the best known network-layer tree-based multicast routing protocols for MANETs. It constructs a loop free shared tree for each multicast group in an on-demand manner.

Tree based approaches are considered to be too fragile for MANETs, which are characterized by frequent node failures and therefore demand more robustness of the protocol. This robustness may be achieved by using redundancy of data forwarding routes.

Each node in the network maintains three routing tables. The first one, simply called Routing Table, records the next hop for unicast routes to other nodes. The second, called Multicast Routing Table, contains entries for the multicast routing groups of which the node is a router. The third, named the Request Table, keeps multicast group addresses and the identifier of the node that made the first route request for that multicast group; this node normally becomes the *group leader*. This last table is maintained by every node in the network and is only used for optimization. If a node later wishes to join a group, it consults this table and discovers the group leader; in case the node has a fresh route to the leader, it may unicast a join request instead of broadcasting it.

A multicast tree is constructed on demand. The first member of the multicast group becomes the leader of the group. This node remains the group leader until it decides to leave the group.

The multicast group leader is responsible for the maintenance and dissemination of the *multicast group sequence number*, a variable that is used to ensure the freshness of the routing information.

When a node has data to send to the multicast group, it broadcasts a Route Request message (RREQ) over the entire network. Furthermore, if the node wishes to join the multicast group it indicates this fact by setting a flag in the RREQ message (called the J-flag). A node receiving a RREQ updates its Routing Table to record the sequence number and next hop information for the source node. This reverse route entry will be used to transmit a reply message back to the source. Only a member of the desired multicast tree may respond to a join RREQ. If the RREQ is not a join request, any node with a fresh route to the multicast group may respond. The responding node updates its Routing and Multicast Routing tables with the information about the next hop to the requesting node's route. Then, it unicasts the reply message (RREP) back to the requesting node. All the nodes along the path add the entry in the Route Table for the node from which they received the RREP creating the forward path.

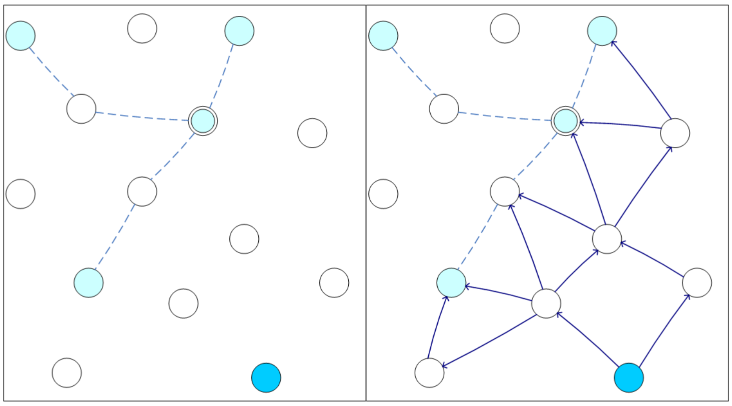
The source node selects the received route with the largest sequence number and the shortest number of hops to the nearest member of the multicast tree. Then it enables the route by unicasting a Multicast Activation message (MACT) to the selected next hop neighbor. The MACT message is propagated to the member of the tree that originated the RREP. All the nodes on the path also become members of the tree. Route activation phase ensures that the multicast tree does not have multiple paths to any node.

The multicast tree construction procedure is depicted in Figure 2.1.

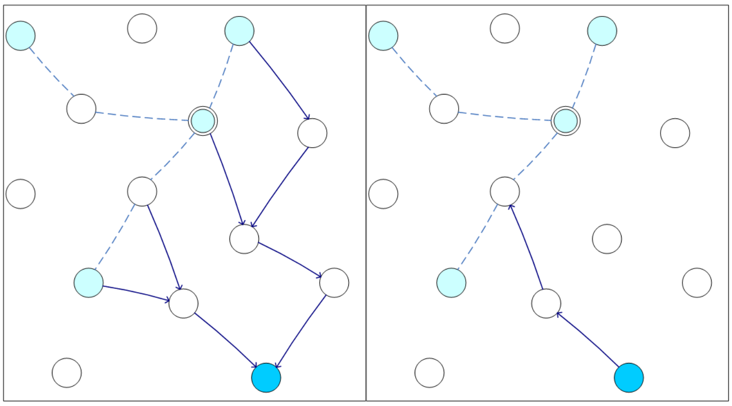
A Group Hello Message is periodically broadcast by the multicast group leader over the whole network to announce its ID and the group sequence number. Upon receiving this message, nodes update their multicast route tables with the group ID, group leader's ID and the sequence number. The Group Hello Message is also used to recover from tree partitioning.

Every member of the tree tracks its tree neighbor that is closest to the group leader. If a failure is detected, a node downstream of the tree initiates the repairing process by broadcasting a special RREQ message. Only the nodes that are at least as close to the leader as the requesting node may respond to this RREQ. This prevents nodes on the same side of the break as the requesting node from responding, thereby ensuring no cycles are formed in the tree.

If, after a predefined number of attempts, no RREP is received, it is assumed that the

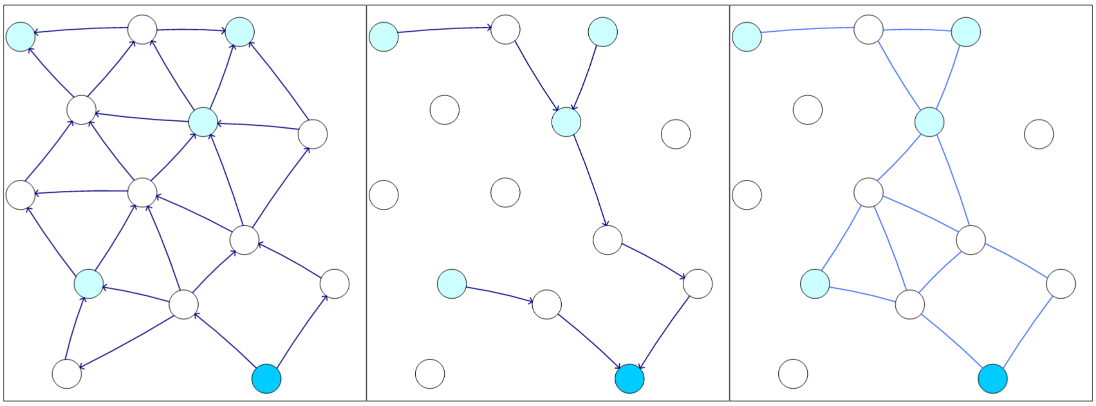


(a) Initial Tree (b) RREQ



(c) RREP (d) MACT

Figure 2.1: MAODV Join Procedure



(a) Join Request (b) Join Reply (c) Resulting Mesh

Figure 2.2: ODMRP Operation

network has become partitioned and the requesting node becomes a new leader of the group. When some tree member node receives a Group Hello Message from another group leader of the same multicast group, the reconnection of the multicast tree is performed.

2.2.2 ODMRP

The *ODRMP - On Demand Multicast Routing Protocol* (Lee, Su, & Gerla 1999) constructs routes from sources to receivers and builds a mesh of nodes, called the *forwarding group*. This protocol exploits the inherent broadcast property of wireless networks. The nodes in the forwarding group do not need to know to whom the message must be forwarded. They simply broadcast the packet to all the one-hop neighbors and only the members of the forwarding group will re-broadcast the message. According to our classification, ODMPR is a reactive network-layer mesh based multicast protocol.

Every multicast sender, while it has data to send to the group, broadcasts periodically to the entire network a JOIN REQUEST message. When a node receives a JOIN REQUEST, it stores a group ID and a next hop to the sender in a Join Table, and then rebroadcasts the message. When the JOIN REQUEST reaches a multicast receiver, the receiver creates or updates a source entry in its Member Table.

Member Tables are periodically broadcast to the neighbors. When a node receives a Member Table it checks if it belongs to its neighbor's routing table. In this case, the node becomes a member of the forwarding group. After route establishment, a multicast source transmits packets broadcasting a message to its neighbors. Only those in a forwarding group will rebroadcast it.

The mesh construction procedure is exemplified in Figure 2.2.

ODMRP is a soft state protocol, meaning that no explicit control packets need to be sent to join or leave a group. If a multicast sender or receiver wishes to leave the group it simply stops refreshing the routes.

2.2.3 PUMA

PUMA - Protocol for Unified Multicasting through Announcements (Vaishampayan & Garcia-Luna-Aceves 2004) is another mesh-based multicast protocol. The protocol uses a single control message, a *multicast announcement* that is exchanged periodically by each network node. One of the purposes of multicast announcements is to elect a core member for the group and to ensure that all nodes in the network have a path to the core. Additionally, all nodes on the shortest paths between any receiver and the core become members of the mesh. Multicast messages are routed to the core until they meet a mesh member; from this point on the messages are flooded in the mesh to reach all multicast receivers.

Each multicast announcement specifies a sequence number, the address of the group (group ID), the address of the core (core ID), the distance to the core, a mesh member flag that is set when the sending node belongs to the mesh, and a parent that states the preferred neighbor to reach the core. With the information contained in such announcements nodes elect cores, determine the routes for sources outside a multicast group to forward data packets towards the group, notify others about joining or leaving the mesh of a group, and maintain the mesh of the group.

The first node that joins the group considers itself a core and starts transmitting periodic (every 3 seconds) multicast announcements incrementing its sequence number. If there are two cores for the same multicast group, the one with a greater core ID wins. Core election is also held in case of network partition.

The nodes in the network, upon receiving a multicast announcement, store the group ID, the sequence number, the core ID, the distance to the core. The node waits for a short period of time to collect multicast announcements from its neighbors and selects one neighbor ID on the shortest path to the core. Then, the node broadcasts the multicast announcement to its neighbors, updating the distance to the core.

Every receiver connects to the elected core along all shortest paths between the receiver and the core. To perform a join procedure, the receiver verifies if it has previously received a multicast announcement for the group. If so, it broadcasts a multicast announcement to its immediate neighbors, setting the mesh member flag. The neighbors one hop closer to the core also consider themselves mesh members and, in turn, broadcast multicast announcements to their neighbors. In this way, nodes on the shortest paths between any receiver and the core collectively form the mesh. If the receiver has never seen any multicast announcement of the group, it considers itself a group core and start transmitting periodic multicast announcements.

If the node wishes to leave a multicast group, it unsets a mesh member flag in its multicast announcements. If the mesh members do not detect mesh neighbors downstream from the core during 2 consecutive multicast announcement periods, they leave the mesh and unset a mesh member flag in their multicast announcements.

A source sends a data packet to its preferred neighbor on the shortest path to the core which, in turn, sends the packet to its own preferred neighbor. When the data packet reaches a mesh member, it is forwarded within the mesh.

Nodes also maintain a packet ID cache to drop duplicate data packets.

2.2.4 ADMR

The periodic signaling used by some structured protocols may substantially limit the benefits of the protocol on-demand operation. The *ADMR - Adaptive Demand-Driven Multicast Routing Protocol* (Jetcheva & Johnson 2001) attempts to reduce as much as possible any non-on-demand components of the multicast protocol.

ADMR uses a routing mesh creation process similar to ODMRP, with the difference that a forwarding group is formed per sender instead of per group. Every source floods the network with its first data packet and each receiver responds with a Receiver Join packet which sets up forwarding paths towards the source.

The route refreshing procedure of ODMRP is replaced by the following mechanism. To each multicast packet, the source node adds a header that contains an approximate time interval in which new packets should be expected. If the source node does not have any data to send to the group, the protocol starts disseminating keep-alive messages to the group. The time between two successive keep-alives is increased by a multiplicative factor. After some period of time, if a node still has no data to send, the keep-alives are stopped and all forwarding state for this sender will expire.

Absence of data packets or keep-alive packets within the specified inter-packet time is an indication of the mesh disconnection. A node that detects a link break is necessarily downstream of the fault. It sends a Repair Notification message to the part of the mesh it is connected and waits some predefined period of time. If no other Repair Notification has been received, it means that the node is the closest to the break, so it initiates a reconnection procedure. A Reconnect message is broadcast to the network with a limited TTL. If some member of the mesh that has not heard any Repair Notification receives a Reconnect message, it assumes that it is upstream of the break and unicasts a Reconnect message to the source node. The Reconnect Reply message is unicast back to the repair node along the path the Reconnect took to reach the source. Each node in the path becomes a forwarder for this multicast source.

If a new receiver wants to join the multicast group, it floods the network with a Multicast Solicitation message. Source nodes respond by unicasting a keep-alive message back to the new receiver. The nodes on the path to the receiver become forwarders for this multicast source.

Unfortunately, in ADMR every source node periodically floods the network with multicast messages. This is done to recover from possible Receiver Join losses. The authors argue that this procedure may be performed in background using a small rate and that it does not introduce much overhead.

A high number of rejoins indicates that the protocol cannot cope with high mobility and the operating mode is switched to flooding. After some period of time, ADMR reverts back to its normal operation, as mobility in the network may have decreased.

2.2.5 AMRoute

Reorganization of routing structures in MANETs is more frequent as compared to fixed networks, since the multicast protocols have to respond to network dynamics in addition to group dynamics. The costs associated with the re-organization may be circumvented by creating an overlay structure that only involves group members, and relying on the underlying unicast protocol to deal with network topology changes. AMRoute (Xie, Talpade, Mcauley, & Liu 2002) was one of the first overlay multicast protocols to be proposed for mobile environments. It is a hybrid between virtual mesh based and tree based approaches.

AMRoute assumes the existence of an underlying unicast routing protocol. An overlay structure is constructed on demand. First, the algorithm constructs a virtual mesh of group members, a graph where each node is a member of the group and every link is a bidirectional unicast tunnel.

In AMRoute, each group has at least one logical core that is responsible for initiating signaling actions: mesh joins and multicast tree creation. Every node begins with identifying itself as the core of a 1-node mesh and broadcasts JOIN_REQ packets with increasing TTL to discover other members of the group. When a member node receives a JOIN_REQ from a core of a different mesh for the same group, the node responds back with a JOIN_ACK . Two meshes merge and a new bidirectional tunnel is established between the member nodes. One of the cores will emerge as the “winning” by a deterministic core resolution protocol.

Subsequently, the protocol creates an overlay tree using unicast tunnels among the member nodes. This procedure is performed the following way. The core sends out periodic TREE_CREATE messages along the links incident on it in the mesh. Group members receiving non-duplicate TREE_CREATEs forward them on all the mesh links except the incoming, and mark the incoming and outgoing links as tree links. If a link is not going to be used as part of the tree, the TREE_CREATE message is discarded and a TREE_NAK is sent back along the incoming links which are then marked as mesh links and not tree links.

The tree topology does not change even if an underlying network topology changes. Thus, AMRoute introduces little control overhead. But, on the other hand, as network topology evolves due to mobility, the costs of data forwarding through overlay links may increase significantly. According to the classification introduced in the previous section, AMRoute is an overlay reactive hybrid multicast protocol.

2.2.6 Adaptive Backbone-Based Multicast

Most well known multicast protocols propose different approaches based on different assumptions about the environment and usually they perform well only under specific conditions. However, there has also been made attempts to develop hybrid multicast protocols that would adjust their behavior dynamically according to current nodes state and network conditions.

An example is *ADB - Adaptive Backbone-Based Multicast* described in (Jaikaeo & Shen 2002). The protocol creates a “forest” of varying-depth trees. The roots of those trees (cores) form a *backbone* which, according to the authors, is a set of nodes that have routes to each other. A core selection process is local to every node and is based on the stability metrics of the node’s neighbors.

Initially, every node sets itself to be a core and sends a Hello message to its neighbors. After some period of time, when the node receives Hello messages from other nodes, it calculates its *height* value, which is a form of stability metric. The height value may be calculated based on link failure frequency, remaining power or degree of connectivity. Based on the height value, the node will decide if it should remain a core or become a child of another node with a better height value. If the node becomes a child of another node, a branch of the local tree rooted on a core is created. The permitted height of the trees depends on local mobility conditions. It means that more nodes will be cores and less tree branches will be formed under high mobility conditions.

The routes between the cores are updated by every node in the network periodically broadcasting its routing tables to the neighbors. Eventually, every core will have knowledge of the shortest paths to other cores. Members of the tree structure also update the routes by periodically sending Hello messages towards the root.

A member of a tree wishing to multicast a packet first sends it to the root which forwards the message to all the backbone nodes that in turn transmit the packet downstream of their trees.

The authors claim that tree structures would be formed in more static areas and, in dynamic zones, a flooding technique would be used. But, as described in the paper, the backbone nodes use unicast to disseminate messages between them. All the network nodes should participate in maintaining the routes between the backbone nodes. That introduces additional overhead in order to gather the information that, in the presence of high mobility, will constantly become stale. Thus, the problem of effective message dissemination in dynamic environments persists.

2.2.7 Random Walk Based Multicast

Deterministic protocols establish exact routes to every member in the group with significant overhead. Due to the network congestion, link failures and nodes movement, still not every member can get all the messages. Hence, different probabilistic unstructured approaches may also be considered in the design of a multicast protocol.

One possible probabilistic method to implement multicast is by using *Random walk* mechanisms that consist in randomly retransmitting a "token" from a node to a randomly chosen neighbor. In this approach, no structure is built and no routing information must be maintained by neither multicast group members, nor relay nodes.

Using random walks for implementing group communication is suggested in (Dolev, Schiller, & Welch 2006). The system design is based on a mobile agent, collecting and distributing information, during a random walk. This mechanism requires low control overhead and network resource consumption to perform data dissemination. But, on the other hand, the message delivery latency introduced by a random walk approach is the major concern. If a node wishes to multicast a message, it has to wait for the agent's arrival. Also, every group member will only receive messages addressed to the group when is visited by the agent.

2.3 Discussion

All the multicast algorithms presented above have advantages and drawbacks.

Flooding is not resource efficient in the general case since all nodes, even those not interested in the multicast, are involved in the data dissemination process. On the other hand, the inherent redundancy of flooding also brings advantages. In particular, flooding is very robust and single failures usually do not compromise data dissemination. Also, flooding is almost oblivious to the topology, and the amount of maintenance operations that it requires in face of node movement is minimal.

Contrarily to flooding, tree based solutions such as MAODV attempt to minimize the number of nodes that participate in the dissemination of data, building close to optimal routes among the multicast group members. Unfortunately, a tree structure is very fragile and can be easily disrupted by the failure or movement of tree members. Therefore, tree-based solutions perform poorly in highly mobile environments.

Tree based solutions may be slightly optimized to cope better with node movement. For instance, MAODV prefers the shortest paths between two members, but that is not necessarily the best solution as, in some situations, longer but more stable paths may perform better. In any case, tree-based protocols start to induce significant overhead when the network is unstable and may also have non-negligible overhead in stable conditions. Again, using MAODV as an example, the fact that Group Hello Messages and RREQs need to be broadcast in the entire system is a major source of overhead and a significant impairment to the protocol stability.

Mesh based solutions are designed to be more robust than a tree based multicast. The robustness is achieved by path redundancy. Thus, this robustness does not come for free. In a mesh-based protocol, such as ODMRP, every source is periodically flooding the network with JOIN REQUEST messages. It has been demonstrated (Kunz & Cheng 2002) that, as the number of senders increases, more network-wide broadcasts are produced and the data delivery ratio drops significantly, due to the broadcast storm phenomenon. In static networks, ODMRP introduces high overheads due to constant broadcasts not only by route refreshing procedures, but also by redundant routes between the multicast group members. Also, in highly dynamic MANETs, the delay between failure detection and new route discovery depends on the frequency of route refreshments. If this procedure is too frequent, the network may become congested.

ADMR is a mesh based approach that seeks to reduce the overhead of periodic broadcasts of control messages, but, on the other hand, that is done by increasing the timing between the failure and its detection, that, in highly mobile environments, may penalize the reliability of data dissemination. This effect is minimized by creating routes from every source to the receivers, again at the cost of disseminating more control messages.

PUMA, another mesh based solution, offers a less expensive mesh construction and maintenance procedure without increasing mesh refreshment delay. However, as will be seen later in this dissertation, if the network, or part of it, contains highly mobile nodes, the protocol, due to its structured nature, does not manage to guarantee high delivery ratios.

Contrarily to mesh based approaches, a random-walk based multicast does not introduce much control overhead, but is extremely sensitive to node failures. If a node fails before retransmitting a token, the information contained in the token may be lost. Also, as mentioned above, this approach suffers from high delays in message dissemination.

Network level multicast requires all nodes run the protocol, while in an overlay approach only the members of the multicast group are required to participate in the protocol. On the other hand, a problem of overlay multicast methods, such as AMRoute, is that the relatively static upper layer may cause redundancy in data delivery in the presence of changes in the underlying topology. Despite this limitation, an overlay solution is good in cases where few nodes take part in the protocol and not every node has to execute it.

Neither of the above protocols exploits the heterogeneity of the network mobility. In a realistic scenario, there will exist more stable parts of the network and some other nodes will be more dynamic. ADB tried to provide an adaptive solution that would tackle this problem. However, we concluded that the protocol does not resolve the problem of mobility: nodes gather and use routing information that quickly becomes stale under high mobility conditions. Another problem of the protocol is the high overhead induced on the core nodes, which also limits the scalability of the algorithm.

Hence, devising an efficient yet robust multicast protocol that does not make any assumptions about network's mobility beforehand still remains a challenging research topic.

Summary

This Chapter has surveyed the main approaches to implement multicast in MANETs. We have seen that these can be classified in two main categories: unstructured and structured approaches. None of these protocols is able to excel in scenarios that have heterogeneous mobility patterns. The next Chapter introduces a protocol that addresses this gap.

3

HAMP – Heterogeneity Aware Multicast Protocol

As discussed in the previous Chapter, most multicast protocols assume some mobility pattern and make the tradeoff between efficiency and reliability at design time. In ad hoc networks that exhibit some stability, it is worth to maintain routes to support the multicast. However, in very dynamic systems, routes become unstable and this forces the structured protocols to rebuild routes, which is an expensive operation that often requires some form of flooding. However, in highly dynamic environments, a repair operation may be too slow to cope with the network dynamism. Furthermore, repair operations may be extremely costly. Therefore, it may be worth to just rely on some form of flooding to support the multicast operation. In this case, routes need only to be rebuilt when some stability is again observed. Thus, the network resources are spent on assuring high message delivery ratios under unstable conditions while improving efficiency whenever the network stabilizes.

We now propose Heterogeneity-Aware Multicast Protocol (HAMP), a multicast protocol that exhibits good performance in environments that have a combination of stable and highly dynamic regions. It combines the use of a mesh-based approach in stable network regions and a flood-based approach in dynamic regions. Both approaches may co-exist in the same network and a seamless transition from mesh-based to flood-based operation is supported by the protocol. In fact, the ad hoc domain may be composed of multiple stable and unstable regions and multicast data propagation may switch multiple times between mesh-based and flood-based forwarding.

3.1 Rationale

Realistic MANET deployments will exhibit heterogeneous behavior along the network. Where the network is reasonably stable, message propagation may be performed in a structured, efficient way, while under high mobility conditions structured techniques may turn out to be unreliable and too expensive to operate. In this situation, the multicast protocol cannot rely on any routing information and should perform data dissemination by using some form of scoped flooding.

Therefore, the operation of nodes in multicast protocols for heterogeneous networks should not be statically configured, based on *a priory* assumptions about the behavior of each individual node. On the contrary, the protocol should be able to observe nodes mobility conditions and adapt the forwarding mode dynamically. Based on this premise we have designed HAMP, a multicast protocol that adapts its dissemination strategy dynamically.

In HAMP, every node observes its neighbors and, using a *stability condition*, verifies if its neighborhood is stable. If so, the node forwards multicast traffic in a structured mesh-mode. Otherwise, the node changes its operation mode to localized flooding, continuing to track the mobility conditions in the neighborhood and reverting to a mesh-operation mode as soon as the network stabilizes.

Summarizing the main idea, HAMP seeks to track nodes' mobility behavior and decides, on a local basis, which is the preferred mode to forward multicast messages. That is done dynamically in order to capture a constantly changing nature of a realistic MANET environment. In the following sections, we present a detailed description of the protocol.

3.2 Building Blocks

HAMP combines a mesh-based multicast protocol and an optimized localized flooding protocol. As will be discussed, one of the main challenges of implementing such combined protocol is how to identify the points in the network where the forwarding strategy must switch from mesh-based to flooding and back to mesh-based, such that reliability is ensured without polluting the entire network with flooding by keeping flooding localized. As building blocks for HAMP we use PUMA (see Section 2.2.3) for structured operation in stable areas of the network and PAMPA (see Section 2.1.1.3) to cross unstable regions.

3.2.1 Structured Approach for Stable Areas

The reason for selecting a mesh-based instead of a pure tree-based multicast algorithm is that a routing mesh has the potential to be more robust than a tree. Therefore, mesh-based solutions are expected to perform better in scenarios with hybrid mobility patterns. Still, as we will see in the experimental section, even a mesh-based solution is not adequate for heterogeneous scenarios. From the multiple mesh-based solutions that can be found in the literature, we have selected PUMA due to its low signaling cost, even in mobile settings.

As described earlier, senders that are not part of the mesh send packets using unicast towards the core until it reaches mesh members. Unfortunately, this procedure is not robust in face of high mobility. Thus, for the sake of simplicity, in the design of HAMP, we require senders to become members of the group. The elimination of this requirement is left for future work.

3.2.2 Scoped Flooding for Highly Mobile Areas

PAMPA is a distance-based flooding scheme that uses the receiving power to estimate the distance to the source and uses this metric to sort the receiving nodes such that nodes more distant from the sender are more likely to retransmit first (in practice, nodes delay the retransmission by an amount of time that is proportional to the measured signal strength). The rebroadcasting is canceled if, during the delay period, a retransmission of the same message is heard c times. That will prevent nodes providing a small additional coverage from retransmitting.

A significant advantage of PAMPA is that there is a strong correlation between the TTL of the flooding and the geographical coverage of the scoped flood. Thus, this makes it easier to limit flooding to a region of the space where there is high mobility (for instance, a street) regardless of the node density in that area.

3.3 HAMP Operation

In a static network, the operation of HAMP approximates the operation of PUMA. Thus, if the network is connected, a single core node will be elected and periodically propagate announcement messages that are used to define a mesh connecting group members. As we explain below, a HAMP multicast announcement carries additional information that is later used to identify *floodgate nodes*, or *FNs*, where message forwarding is switched from mesh-based to flooding and vice-versa.

As a result of the PUMA operation, each node in the network is able to retrieve the following information:

- if the node is a member of the group mesh;
- which neighbor(s) lay on the shortest route to the core;
- what is the distance to the core and if there are further mesh members downstream from the core.

Based on the periodic exchange of PUMA's multicast announcements, a node can detect the presence of instability in the network if one or both of the two following situations occur:

- the upstream mesh-members, nodes on the shortest route(s) to the core of the group, are unstable;
- the downstream mesh members are unstable.

In the following section, we discuss the stability condition implemented in HAMP.

3.3.1 Stability Condition

The notion of stability is closely related to mobility. Nevertheless, even in presence of mobility, as long as the mesh remains connected and can forward multicast traffic in a structured way, the protocol should avoid the use of flooding as a forwarding strategy. On the other hand, if the nodes that form a mesh change too frequently, there is a strong probability that the mesh may become broken, and even temporary mesh disconnections may cause loss of data packets and compromise the protocol robustness. Therefore, we characterize the ability of the protocol to rely on structured mesh based forwarding in terms of stability of mesh members.

We recall that the PUMA algorithm creates a tree-like routing mesh. As long as the nodes of each tree branch have a route to the core, the mesh remains connected. The nodes determine who are their mesh parents, their mesh neighbors on the path to the core, based on the information about the distance to the core exchanged in multicast announcements.

Still based on the multicast announcements, the node determines if it has a stable route to the core: if there is at least one parent in the neighborhood that is present during k consecutive multicast announcements, the node is considered stable.

If the condition of at least one unchanging upstream node is not held, the neighborhood is considered unstable. The following aspects should be emphasized regarding this notion of stability:

- Stability is a local property, i.e. the network may include multiple stable and unstable regions. The protocol should be able to identify these zones and adapt its behavior in each of them accordingly;
- Furthermore, the node on the boundary may have the instability zone either upstream or downstream from the core. That should be considered when deciding if the node must switch between the routing modes according to the direction from where a multicast message is received.

As noted, the idea behind HAMP is that unstable nodes should perform a localized flooding while, stable nodes continue executing a mesh-based forwarding. The challenge is to identify which nodes should switch between mesh and flood based forwarding. Another challenging task is computing the adequate TTL to be used when a scoped flooding is initiated.

The stability detection procedure is depicted in Algorithm 1. A node first verifies who are its neighbors and compares the result with the observed neighborhood in the previous interval. If there are no common neighbors (lines 2–7), the node considers itself unstable and becomes a *floodgate node*, adjusting its *flood horizon* (we will later describe how this value is computed), and sends an instability announcement (lines 8–12).

3.3.2 Electing Floodgate Nodes

We designate *floodgate node* (FN) a node that is in charge of changing the forwarding mode of a multicast message, from mesh-based to flood-based and vice-versa. These nodes will be located on the boundary between the stable and unstable regions of the network. If the unstable region is downstream from the FN (with regard to the core), the node is called an *upstream* floodgate node, or UFN. Conversely, if the flood region is upstream from the FN, the node is

Algorithm 1: HAMP Protocol – Stability Verification

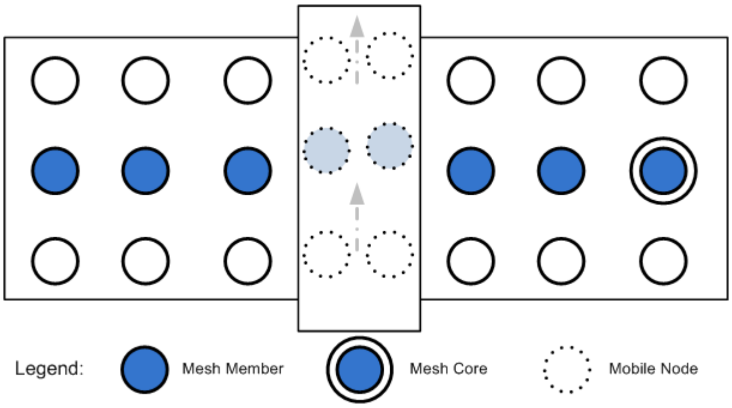
```
VerifyStability
input: group
stable[group ] $\leftarrow$ False;
foreach parent $\in$ parents(group) do
    if parent $\in$ oldParents[group ] then
        stable[group ] $\leftarrow$ True;
    end
end
if not stable[group ] then
    FN[group ] $\leftarrow$ True;
    FH[group ] $\leftarrow$ distanceToStable[group ];
    SendInstabilityAnnouncement(group);
end
else if not heardInstability[group ] then
    FN[group ] $\leftarrow$ False;
end
oldParents[group ] $\leftarrow$ parents(group);
heardInstability[group ] $\leftarrow$ False;
```

called a *downstream* floodgate node, denoted DFN. Floodgate nodes have an associated attribute called *flood horizon* (FH) which corresponds to the TTL value that must be associated with any scoped flood initiated by a FN.

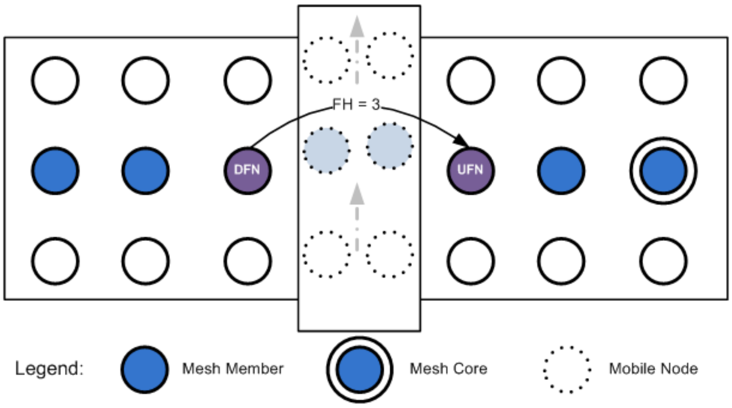
In order to allow the identification of FNs we enriched the information that is propagated by PUMA announcements so that it captures the route from the group core to other mesh nodes in the network. Note that, if the network region between a node n and the core is stable, most times the node n will receive announcements from the core via the same route. On the other hand, if there is an unstable region between n and the core, (part of) that path will be changing frequently.

In HAMP, downstream floodgate nodes are determined first. Then, these nodes are responsible for identifying appropriate upstream floodgate nodes. This process is illustrated in Figure 3.1. A node n elects itself as a downstream floodgate node when all the following conditions are verified:

- n is a mesh member;
- all its upstream neighbors have changed since last k multicast announcements;
- its downstream neighbors are believed to be stable, i.e. n has not been notified about instability by the downstream nodes, or it has no downstream neighbors.



(a) Initial Configuration



(b) After Election

Figure 3.1: Floodgate Nodes Election

After electing itself as DFN, node n must calculate its *flood horizon* (FH_n). To that end, the node n compares the last m paths to the core received from its upstream mesh neighbors and identifies the closest node u that appears in most of these paths (u is very likely to be a stable node). Then, node n sets FH_n to the distance between n and u that corresponds to the minimum number of hops to u in the paths. Note that there is always one stable node u in the paths: in the worst case, the core is considered the closest stable node.

After electing itself as a DFN and setting its flood horizon, node n proceeds to notifying the upstream nodes about instability attempting to reach a UFN (or the core, if there is no stable mesh members closer to n). For that purpose, it initiates a scoped flood of an INSTABILITYANNOUNCEMENT message with $TTL = FH_n$. This message includes the following fields: the identifier of the source DFN and the distance, in number of hops, of the source node to the core (denoted D_{nc}) and n's flood horizon, or FH_n.

Any mesh node j that receives an INSTABILITYANNOUNCEMENT sets its state to upstream floodgate node, if and only if, the following conditions hold:

- j is a mesh member;
- it has at least one upstream neighbor that has not changed for the last k multicast announcements;
- its distance to the core D_{jc} is smaller than D_{nc}.

In this case, it computes the maximum distance to all DFNs received during the last and current k consecutive multicast announcement periods and stores this value as the node's FH. Also, as for the upstream floodgate node the DFN that generated the INSTABILITYANNOUNCEMENT is its known downstream neighbor, UFN continues to consider itself a mesh member for at least next k multicast announcements. This mechanism ensures that UFNs do not leave the mesh due to instability of their immediate downstream mesh members.

Note that, in response to an INSTABILITYANNOUNCEMENT sent by a given DFN, multiple mesh members that lie upstream of that DFN may become UFNs. This is not a problem, as the flooding procedure avoids the propagation of duplicates.

3.3.3 Demoting Floodgate Nodes

Given that the network is dynamic, it is possible that an unstable region may stabilize at a later time. In this case, flooding must revert to mesh based routing. Thus, the floodgate status is not permanent.

A downstream floodgate node continues to check its stability every k announcement periods and, if the conditions for being considered DFN (described earlier in this section) are not met, the DFN stops refreshing its status with InstabilityAnnouncements.

On the other hand, upstream floodgate status is treated as softstate. That means that if an UFN has not heard any InstabilityAnnouncement in the last k announcements period, it assumes that the network downstream stabilized and demotes itself from the status of UFN reverting to normal mesh forwarding.

The promotion and demotion of floodgates is determined during the production and reception of instability announcements. These procedures are captured in detail by Algorithm 2. When instability in the neighborhood is detected, a node produces an InstabilityAnnouncement (lines 1–5). A receiving node decrements the announcement's TTL (line 9) and, if it is a member of the corresponding group, adjusts its floodgate status and flood horizon according to the criteria previously described (lines 10–18). The announcement is further propagated until the TTL reaches zero (lines 19–21).

3.3.4 Data Forwarding

The data forwarding functions are described in Algorithm 3 and work as follows:

- If a mesh member is not a floodgate, it forwards messages received in *mesh-mode* using the regular PUMA rules, i.e. these messages are retransmitted if the node has other mesh members in the direction the message is traveling (lines 17–19).
- If a UFN node j receives a message in *mesh-mode* from an upstream node, it retransmits the message in *flood-mode* with a $TTL = FH_j$ (computed as described before) indicating the UFN's distance to the core and that the flooding direction is downstream. Symmetrically, if a DFN node j receives a message in *mesh-mode* from a downstream node, it retransmits

Algorithm 2: HAMP Protocol – Instability Announcements

```
1  SendInstabilityAnnouncement
   input: group
2  pkt.DistanceToCore $\leftarrow$ distanceToCore(group);
3  pkt.TTL $\leftarrow$ FH[group ];
4  pkt.Mode $\leftarrow$ FLOOD;
5  pkt.OriginalTTL $\leftarrow$ pkt.TTL;
6  send(pkt);

7  ReceiveInstabilityAnnouncement
   input: pkt, group
8  if pkt.ID $\notin$ prevReceivedIDs then
9      prevReceivedIDs $\leftarrow$ prevReceivedIDs $\cup$ pkt.ID;
10     pkt.TTL $\leftarrow$ pkt.TTL $- 1$;
11     if isMeshMember(group) then
12         if isStable(group) and distanceToCore(group) $<$ pkt.DistanceToCore then
13             FN[group ] $\leftarrow$ True;
14             heardInstability[group ] $\leftarrow$ True;
15             dDFN $\leftarrow$ pkt.DistanceToCore $-$ distanceToCore(group);
16             dUFN $\leftarrow$ group.DistanceToCore $-$ (pkt.DistanceToCore $-$ pkt.OriginalTTL);
17             FH[group ] $\leftarrow$ MAX(dDFN,dUFN);
18         end
19     end
20     if pkt.TTL $> 0$ then
21         send(pkt);
22     end
23 end
```

the message in *flood-mode* with $TTL = FH_j$ indicating the DFN's distance to the core node and that the flooding direction is upstream (lines 9–12 and 17–19).

- If a flooded message with $TTL \geq 0$ is received by a regular mesh member, the message is propagated in *mesh-mode*, as long as the node considers itself stable (line 13–19).
- Every other node in the network receiving a message in *flood-mode* and $TTL > 0$ propagates the message using the PAMPA algorithm and decrements the TTL (lines 22–24).
- If a flooded message is received by a non-mesh node with $TTL = 0$ the packet is dropped.

Regardless of the message forwarding mode every node only retransmits a packet when it is received for the first time. To guarantee this condition nodes maintain a cache of the last messages received, in order to be able to detect and discard duplicates.

3.4 Core Stability

HAMP relies on the information about the neighbors on the path to the core in order to identify unstable zones. However, if the core itself is an unstable node the protocol reliability may be affected. Regarding this issue the following aspects of the HAMP protocol should be considered:

- as every node gathers information about its mesh parents, the immediate neighbors of the core will be able to determine the core instability and adapt its flooding horizon accordingly;
- as flooding is spread in all directions there is a strong probability that, as the core moves, it still can be reached by multicast traffic between the updates of the flooding horizon.

The implementation of core migration procedure to a more stable node would have serious disadvantages, as possibly frequent core changes could lead to higher control overhead and inevitable packet losses; in the worst case, if there were no stable group members, the problem of an unstable core would persist in any case.

Summary

In this Chapter, we described HAMP, a Heterogeneity-Aware Multicast Protocol that combines structured and unstructured forwarding approaches. The algorithm tracks local mobility conditions and selects the forwarding technique according to a stability condition evaluated dynamically. The node stability is inferred by periodically checking the list of neighbors in the path to the core. Using this mechanism it is possible to apply mesh-based forwarding in stable regions and scoped flooding in unstable zones. The procedure for verifying the network stability is periodically re-evaluated by the nodes so that if any changes are observed the nodes can adapt their behavior accordingly.

Algorithm 3: HAMP Protocol – Data Sending and Reception

```
ReceiveData
input: pkt, group
if pkt.ID $\notin$ prevReceivedIDs then
    prevReceivedIDs $\leftarrow$ prevReceivedIDs $\cup$ pkt.ID;
    pkt.TTL $\leftarrow$ pkt.TTL $-$ 1;
    if isReceiver(group) then
        accept(pkt);
    end
    if isMeshMember(group) then
        if isFN(group) and pkt.Mode = MESH then
            pkt.TTL $\leftarrow$ FH(group);
            pkt.Mode $\leftarrow$ FLOOD;
        end
        else if isStable(group) and pkt.Mode = FLOOD then
            pkt.TTL $\leftarrow$ maxTTL;
            pkt.Mode $\leftarrow$ MESH;
        end
        if pkt.TTL $> 0$ then
            send(pkt);
        end
    end
    else
        if pkt.Mode = FLOOD and pkt.TTL $> 0$ then
            send(pkt);
        end
    end
end

SendData
input: pkt, group
pkt.DistanceToCore $\leftarrow$ distanceToCore(group);
if isFN(group) then
    pkt.TTL $\leftarrow$ FH[group ];
    pkt.Mode $\leftarrow$ FLOOD;
end
else
    pkt.Mode $\leftarrow$ MESH;
end
send(pkt);
```

4
Evaluation

To validate and evaluate the performance of HAMP, we executed a series of experiments that compare the performance of HAMP against PUMA and PAMPA which are representatives of the state of the art of routing schemes in mobile ad hoc networks. The evaluation compares the reliability and efficiency of the protocols in terms of delivery ratios and number of messages exchanged by all protocols in several scenarios with homogeneous and heterogeneous mobility patterns to highlight the trade-offs involved in the different approaches.

4.1 Experimental Settings

The NS-2 simulator for mobile networks with IEEE 802.11 MAC layer model was used to test the protocols. The PUMA implementation for NS-2 was available from SourceForge [1] and the PAMPA implementation was available from the author's personal website [2]. An implementation of HAMP was developed for NS-2, using the two previous implementations as building blocks.

4.2 Performance and Cost Metrics

In order to compare the algorithms, we used the following metrics.

- *Packet Delivery Ratio.* The ratio between data packets delivered to the destinations and data packets originated by the sources. This metric captures the reliability of a protocol.
- *Number of Total Bytes Transmitted per Data Packet Delivered* is the ratio between the total amount of all bytes transmitted by all nodes in the network and the number of data packets delivered to the destinations. This count includes transmissions of packets that

---

[1] `http://sourceforge.net/projects/puma-adhoc/`

[2] `http://www.di.fc.ul.pt/~hmiranda/pampa/`

are eventually dropped or retransmitted by intermediate nodes. This metric captures the efficiency of protocol operation.

- *Number of control bytes transmitted per data packet delivered* measures the control overhead induced by the protocols.

4.3 Simulation Parameters

The value of the parameter k used in HAMP to define local stability is set to 2 in all tests. All the results presented are averages of runs in ten different scenarios with the same properties.

There is one sender in the network that, every second, sends to the group a data packet with 512 bytes of size. The duration of each experiment is 300 seconds. Group size varies across the experiments between 2, 5, 10, 20 and 40 nodes.

In the following sections we describe the mobility patterns used in the tested scenarios and present and discuss the simulation results for each scenario.

4.4 Grid Scenario

We start by depicting the performance of HAMP against homogeneous protocols in a scenario where nodes are uniformly distributed in three adjacent geographical regions. Figure 4.1 shows a configuration for 10 group members. Zones A (on the left) and C (on the right), have an area of 200×140 square meters each, and have static nodes. They represent, for instance, classrooms, libraries, cafeterias, etc. Zone B (on the center) was placed in the middle of the space and has an area of 40×300 square meters where nodes may be mobile. It represents, for instance, a street or a corridor.

A total number of 110 nodes is deployed in the entire space with the following distribution: 40 nodes in zone A, 40 nodes in zone C and 30 nodes in region B. Within each region, nodes are distributed uniformly in a regular grid. The transmission range is set to 30 meters so that the nodes can only communicate with their immediate neighbors. This allows us to have a network with a reasonable diameter which makes more challenging for the protocols to operate in a more complex environment. Nodes in region B follow a variant of the Manhattan model, where nodes

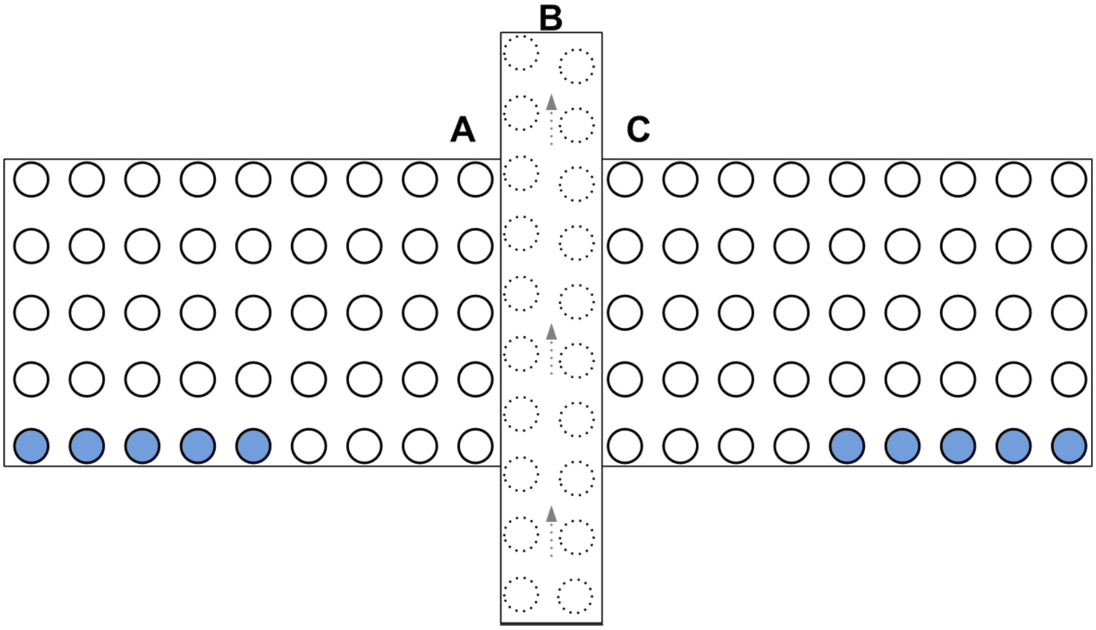


Figure 4.1: Grid

move vertically in one direction until they reach the border of the space, where they start moving in the opposite direction. The speed of nodes in zone B ranges from 0m/s to 50m/s.

For all tests, both multicast receivers and the sender are placed in the stable regions A and C (half of the group in each stable zone).

4.4.1 Packet Delivery Ratio

In Figure 4.2, we illustrate the evaluation results for a test with 20 mesh members. The Table 4.1 presents the results for different group sizes.

Figure 4.2 shows that the delivery ratios of PAMPA, as expected, are the highest due to the resilience of the flooding scheme. On the other hand, the performance of the PUMA protocol is severely affected by the mobility and the delivery ratio drops significantly as the speed of nodes in the mobile region increases. In turn, HAMP, for 20 group members, exhibits delivery ratios very close to the PAMPA results, always close to 95% or higher, exhibiting little sensitivity to changes in the speed of nodes.

Based on more detailed simulation results shown in Table 4.1, we further verify that in all tests HAMP outperforms PUMA when considering the delivery ratio metric. Also, the results in Table 4.1 show that HAMP exhibits little sensitivity to the speed of nodes. As long as the

Table 4.1: Delivery Ratio for Grid Scenario

		Speed						
Nodes		0 m/s	5 m/s	10 m/s	15 m/s	20 m/s	25 m/s	50m/s
2	HAMP	0.9934	0.9698	0.9714	0.9752	0.9672	0.9608	0.9390
	PAMPA	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	0.9920
	PUMA	0.9962	0.8980	0.6848	0.5786	0.5450	0.5058	0.5070
5	HAMP	0.9858	0.9622	0.9702	0.9735	0.9725	0.9634	0.9398
	PAMPA	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	0.9888
	PUMA	0.9988	0.8794	0.6194	0.4950	0.4557	0.4079	0.4067
10	HAMP	0.9906	0.9712	0.9742	0.9753	0.9770	0.9646	0.9574
	PAMPA	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	0.9888
	PUMA	0.9974	0.8976	0.6924	0.5861	0.5528	0.5080	0.5072
20	HAMP	0.9606	0.9621	0.9636	0.9650	0.9594	0.9570	0.9480
	PAMPA	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	0.9884
	PUMA	0.9801	0.9733	0.8690	0.7208	0.6454	0.5853	0.5048
40	HAMP	0.9762	0.9800	0.9829	0.9821	0.9822	0.9763	0.9712
	PAMPA	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	0.9882
	PUMA	0.9765	0.9719	0.9368	0.8501	0.7445	0.7037	0.5691

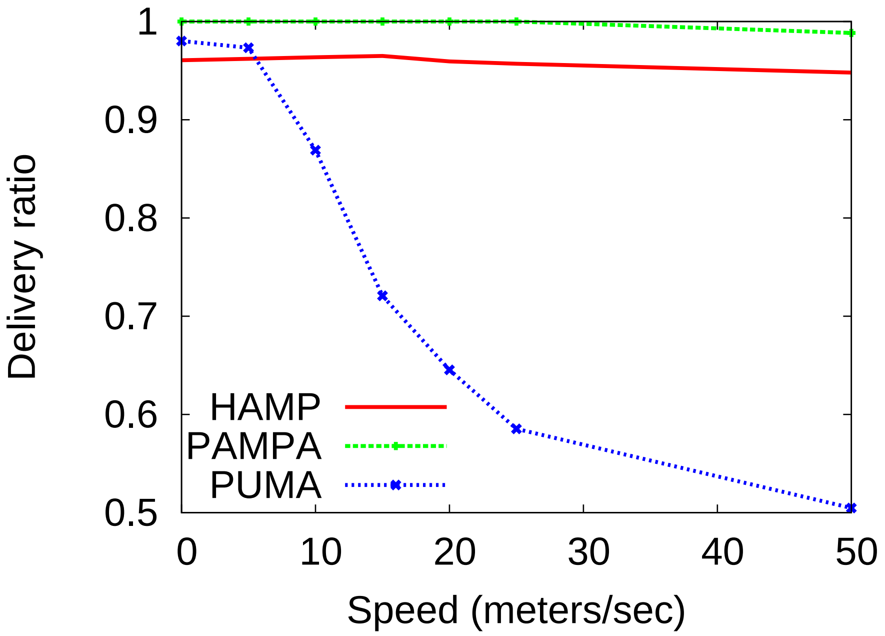


Figure 4.2: Delivery Ratios For Grid Scenario

protocol succeeds to identify unstable zones, a scoped flooding scheme ensures the multicast traffic crosses those areas regardless the speed of the mobile nodes.

Contrarily to HAMP, as the node speed in the mobile zone increases, the performance of PUMA is severely degraded. In fact, under extremely high mobility, 25m/s and 50m/s, data packets do not manage to cross the zone B. For that reason, the delivery ratios achieved by PUMA in these scenarios are about 50%; only half of the group members – those located on the sender's side of the scenario – receive multicast traffic. As mobile mesh members start moving faster, the mesh in those areas becomes disconnected more frequently and the delivery rate drops significantly.

However, for a small group size (of 2 and 5 nodes), the delivery ratios of HAMP are slightly lower than in tests with many group members. This is explained by the fact that the mesh formed on stable regions is much thinner and, consequently, more failure-prone.

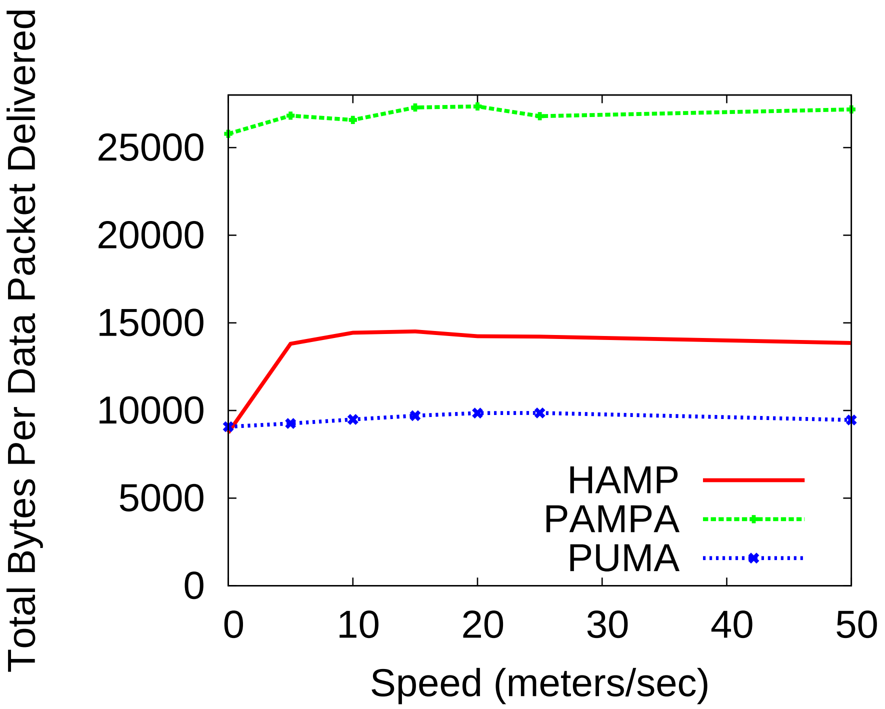


Figure 4.3: Total Overhead For Grid Scenario

4.4.2 Number of Total Bytes Transmitted per Data Packet Delivered

Considering the delivery ratio metric, we confirmed that the flooding protocol is oblivious to the node movement. Unfortunately, as can be observed in Figure 4.3, the total cost to deliver every message is much higher compared to two other protocols. For 20 group members, the total overhead in PAMPA is twice that in HAMP.

The total overhead induced by PUMA is 30% lower than that of HAMP, while the observed delivery ratios of PUMA are about 45% lower in highly mobile settings. This result allows us to argue that the higher overhead of HAMP is justified by significantly higher delivery ratios. On the other hand, the total cost generated by HAMP is much lower than in PAMPA.

In Table 4.2, we see that by commuting to localized flooding in an unstable region of space, and unlike PUMA, HAMP can maintain a high delivery ratio in the heterogeneous mobility scenario, with a much smaller cost than when using flooding in the entire network. In fact, for a scenario with 20 group members, as shown in Figure 4.2, the delivery ratio achieved by HAMP is only about 4% lower than in PAMPA while, as shown in Figure 4.3, the total cost per data

Table 4.2: Total Bytes Sent per Data Packet Delivered for Grid Scenario

		Speed						
Nodes		0 m/s	5 m/s	10 m/s	15 m/s	20 m/s	25 m/s	50m/s
2	HAMP	42261	125687	129328	134795	131812	127748	119325
	PAMPA	257834	268202	265787	272881	273447	267898	270771
	PUMA	48215	49211	50512	50428	48767	48571	47194
5	HAMP	20377	50286	53119	52768	52757	52607	49312
	PAMPA	103133	107281	106315	109152	109378	107159	108659
	PUMA	19378	20342	22331	23573	23866	24277	23456
10	HAMP	10881	25303	26406	26012	26396	26703	24876
	PAMPA	51566	53640	53157	54576	54689	53579	54329
	PUMA	9956	10115	10591	10548	10179	9874	9750
20	HAMP	8078	13857	14228	14490	14213	14266	13667
	PAMPA	25783	26820	26578	27288	27344	26789	27175
	PUMA	9075	9258	9490	9705	9848	9859	9459
40	HAMP	5520	7738	7745	7796	7834	7764	7351
	PAMPA	12891	13410	13289	13644	13672	13394	13590
	PUMA	6851	6914	6924	6940	7044	6994	6833

packet delivered is about a half.

When a group size is very small relatively to the total number of nodes in the network, the cost of the HAMP protocol is significantly higher than the cost of PUMA. However, as a number of group members increases, the advantage of HAMP becomes more visible. And, on the other extreme, for a big group, in HAMP, and also in PAMPA, overhead is even lower. As a mesh size grows, a significant percentage of the nodes in the network form the mesh. In this case, PUMA floods packets through the entire mesh whereas PAMPA applies a more efficient approach. In this case, as HAMP also uses PAMPA's scheme in mobile regions, the total overhead in HAMP is lower than in PUMA.

The simulation results also show that, in HAMP, the total overhead does not increase as the nodes speed increases. The overhead gets higher as mobility is detected and remains uniform, as the protocol identifies the instability regions and maintains the flooding confined to those zones, regardless of node speed in those regions. Consequently, as in this test the area of the unstable zone is fixed, flooding horizons calculated by mesh nodes also remain roughly unchanged regardless the speed of the unstable nodes.

4.4.3 Control Overhead

We now compare the control overhead induced by the structured protocol PUMA and the hybrid approach HAMP. Given that PAMPA does not use control messages, it is not evaluated against this metric.

The simulation results, depicted in Figure 4.4 for 20 group members and presented in more detail in Table 4.3 for the control overhead, show that the additional cost required by HAMP, compared to total cost per packet delivered, is not significant.

In the scenario with no mobility, in HAMP, the control overhead is about 40% higher than in PUMA. Initially, it suffers an additional increase as mobility is detected. However, when the node speed increases, the control overhead does not change significantly given that the size of the mobile region is fixed and scoped flooding is confined to roughly the same neighborhood.

In case of PUMA, the mobility makes the delivery ratio drop severely, and control overhead per every packet delivered becomes higher. For this reason, in cases of high mobility, HAMP shows lower control overhead per every packet delivered than PUMA.

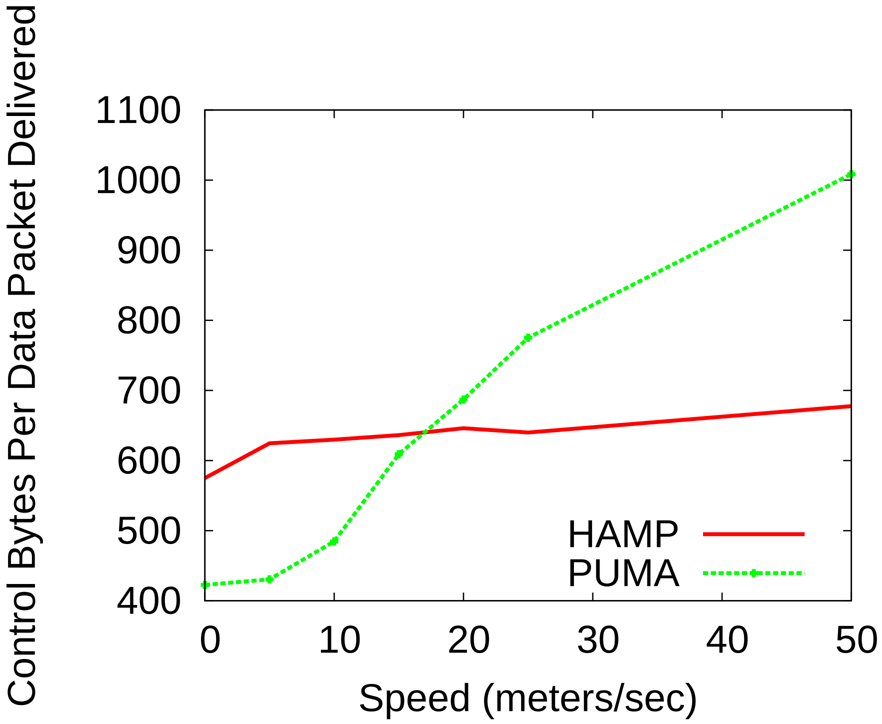


Figure 4.4: Control Overhead For Grid Scenario

Table 4.3 also shows that, as expected, the control overhead per data packet delivered decreases, for both HAMP and PUMA, as the number of group members becomes higher.

4.5 Random-Street-Random Scenario

We proceed the evaluation by testing the protocols in a more realistic scenario where the nodes in the static areas A and C are distributed randomly, as shown in Figure 4.5.

A total of 140 nodes were deployed in this scenario. Zones A and C have areas of 130 × 110 square meters. In order to guarantee the connectivity in static zones with random node placement we increased to 50 the number of nodes deployed in each of these areas. Zone B has an area of 40 × 400 square meters and carries 40 nodes.

The rest of the settings remained unchanged.

Table 4.3: Control Bytes Sent per Data Packet Delivered for Grid Scenario

		Speed						
Nodes		0 m/s	5 m/s	10 m/s	15 m/s	20 m/s	25 m/s	50m/s
2	HAMP	5446	6080	6158	6157	6233	6272	6479
	PUMA	4058	4545	6008	7173	7633	8343	9021
5	HAMP	2211	2460	2496	2469	2493	2516	2650
	PUMA	1637	1881	2691	3353	3716	4184	4493
10	HAMP	1101	1242	1254	1241	1248	1275	1306
	PUMA	822	925	1206	1447	1532	1680	1815
20	HAMP	574	623	628	641	638	645	669
	PUMA	422	430	484	609	687	775	1008
40	HAMP	277	294	294	300	302	306	314
	PUMA	209	216	224	249	295	323	424

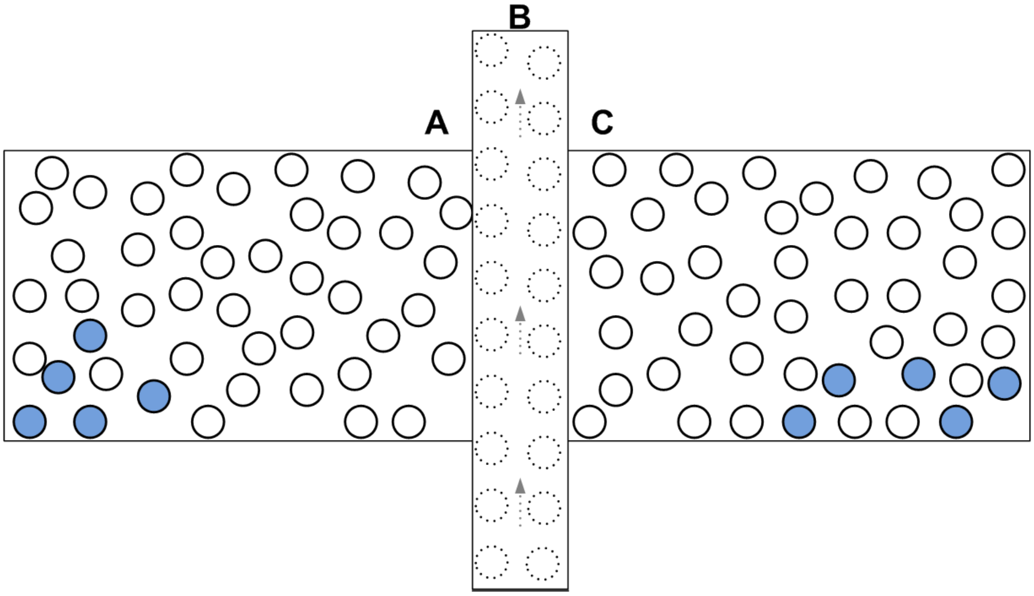


Figure 4.5: Random-Street-Random Scenario

4.5.1 Packet Delivery Ratio

Results for the random scenario, with 20 group members, are depicted in Figure 4.6. As expected, the results are very similar to the ones obtained in the previous scenario. Namely, PAMPA and HAMP demonstrate very high delivery ratios while the robustness of PUMA is significantly affected by the mobility.

Table 4.4 shows the delivery ratios for the evaluated protocols in more detail. As can be noted, the evaluation results for this scenario are consistent with the previous simulations.

Therefore, in these experimental settings, the random placement of the nodes does not affect the robustness of HAMP. The delivery ratios achieved by HAMP are even higher than in the previous scenario. This is explained by the fact that a random placement of the nodes in this scenario that originates a thicker, and consequently more robust, mesh structure. On the other hand, PUMA, as in the previous scenario, does not succeed to cross the mobile zones and fails to deliver multicast traffic to the group members in the static zone opposite to the multicast sender.

Table 4.4: Delivery Ratio for Random-Street-Random Scenario

		Speed						
Nodes		0 m/s	5 m/s	10 m/s	15 m/s	20 m/s	25 m/s	50m/s
2	HAMP	0.9830	0.9776	0.9774	0.9764	0.9726	0.9642	0.9220
	PAMPA	1.0000	0.9980	0.9998	0.9992	0.9986	0.9994	0.9806
	PUMA	0.9822	0.9418	0.8156	0.7598	0.6644	0.5892	0.5130
5	HAMP	0.9649	0.9706	0.9733	0.9728	0.9675	0.9520	0.9398
	PAMPA	1.0000	0.9972	0.9991	0.9988	0.9978	0.9991	0.9755
	PUMA	0.9670	0.9278	0.7971	0.7317	0.6254	0.5295	0.4198
10	HAMP	0.9626	0.9756	0.9768	0.9702	0.9779	0.9577	0.9474
	PAMPA	0.9903	0.9966	0.9993	0.9987	0.9977	0.9992	0.9778
	PUMA	0.9570	0.9485	0.8397	0.7944	0.7175	0.6132	0.5161
20	HAMP	0.9713	0.9778	0.9798	0.9766	0.9743	0.9632	0.9511
	PAMPA	0.9951	0.9965	0.9990	0.9984	0.9977	0.9991	0.9741
	PUMA	0.9706	0.9427	0.8660	0.8091	0.7213	0.6190	0.5143
40	HAMP	0.9812	0.9741	0.9834	0.9787	0.9850	0.9739	0.9704
	PAMPA	0.9876	0.9961	0.9985	0.9984	0.9939	0.9991	0.9741
	PUMA	0.9703	0.9385	0.9117	0.8402	0.7926	0.7086	0.5529

Table 4.5: Total Bytes Sent per Data Packet Delivered for Random-Street-Random Scenario

		Speed						
Nodes		0 m/s	5 m/s	10 m/s	15 m/s	20 m/s	25 m/s	50m/s
2	HAMP	59165	127714	143962	151231	151477	145956	139303
	PAMPA	214524	223516	232007	224165	230114	230267	219995
	PUMA	44766	56671	69614	78087	79017	68909	70181
5	HAMP	30755	54168	60372	61785	63138	62177	62065
	PAMPA	85809	89514	92765	89630	92082	92070	88431
	PUMA	25818	32168	37973	39271	44880	41939	43782
10	HAMP	16381	29032	30656	31636	32477	31078	32041
	PAMPA	43338	44775	46345	44815	46041	46016	44107
	PUMA	18041	20488	22137	23160	25475	22898	25488
20	HAMP	9730	15305	16001	16670	16297	15612	16630
	PAMPA	21560	22387	23177	22407	23020	23003	22135
	PUMA	12075	12959	13216	13634	14111	13062	14726
40	HAMP	6687	8628	8894	8942	9080	9043	9122
	PAMPA	10860	11198	11595	11204	11554	11501	11067
	PUMA	9907	10307	9953	10148	10866	10415	11150

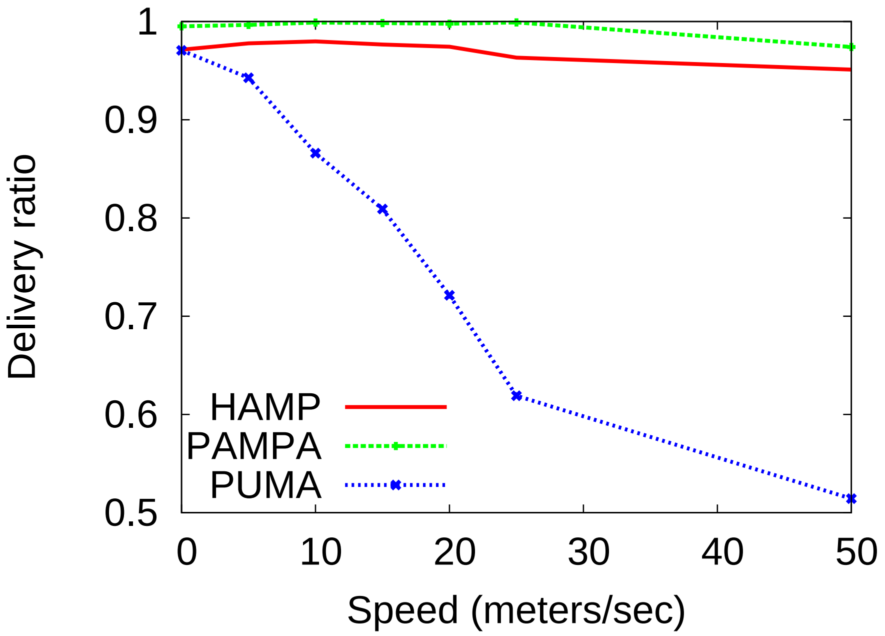


Figure 4.6: Delivery Ratios For Random-Street-Random Scenario

4.5.2 Number of Total Bytes Transmitted per Data Packet Delivered

The total overhead for the Random-Street-Random scenario is illustrated in Table 4.5. Figure 4.7 depicts the results for 20 mesh members.

While the delivery ratios and, as will be shown next, the control overhead per data packet delivered are similar to the ones obtained in the previous scenario, the total cost to deliver a multicast message, for both PUMA and HAMP, are higher than in a grid scenario. Once again, this is explained by the random positioning of the nodes that makes a mesh size grow and more nodes participate in data forwarding that, in turn, leads to higher costs to deliver every multicast packet.

4.5.3 Control Overhead

Figure 4.8 shows that the control overhead induced by PUMA and HAMP follows the same pattern as in the previous scenario. It starts with a lower overhead for PUMA in tests with low mobility. But, as the node speed increases, HAMP offers lower control overhead per data packet

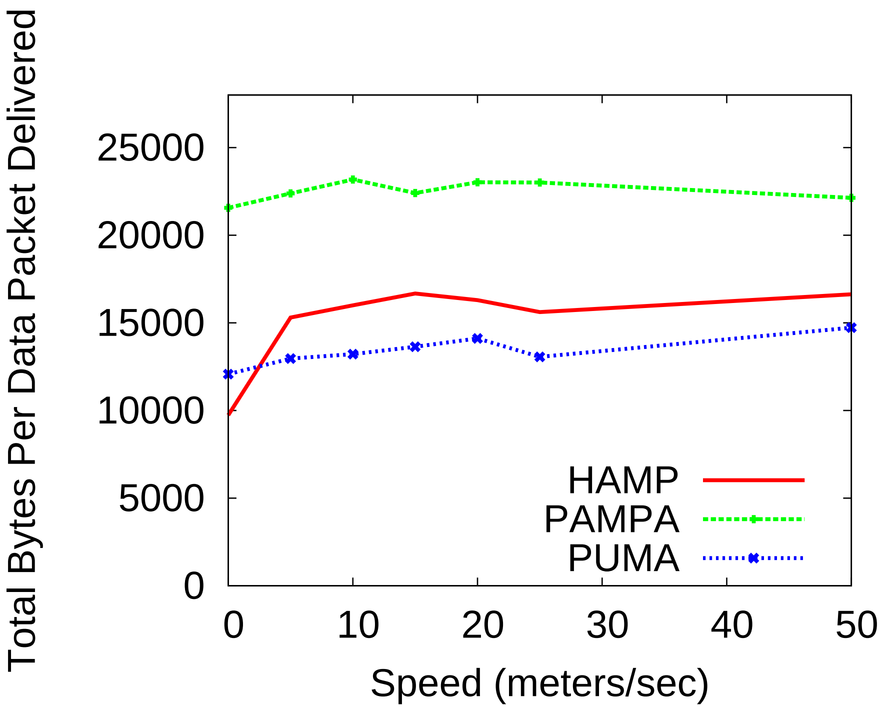


Figure 4.7: Total Overhead For Random-Street-Random Scenario

delivered, due to the higher delivery ratios achieved.

Table 4.6 illustrates the control overhead for different group sizes in this scenario.

In general, the simulation results for this scenario further show that HAMP adapts much better to such heterogeneous networks, and offers a very favorable trade-off between efficiency and robustness compared to a pure structured and unstructured approaches.

4.6 Scenario With An Unstable Core

In Chapter 3, we have discussed the operation of HAMP in a scenario where the core is placed in an unstable zone. We argued that the core's mobility does not compromise the protocol's robustness and, even in these situations, the HAMP protocol will still manage to demonstrate high delivery ratios.

To illustrate that, we performed tests in a scenario with the previous configurations, only changing the location of the group core node and placing it in the unstable zone. It is worth

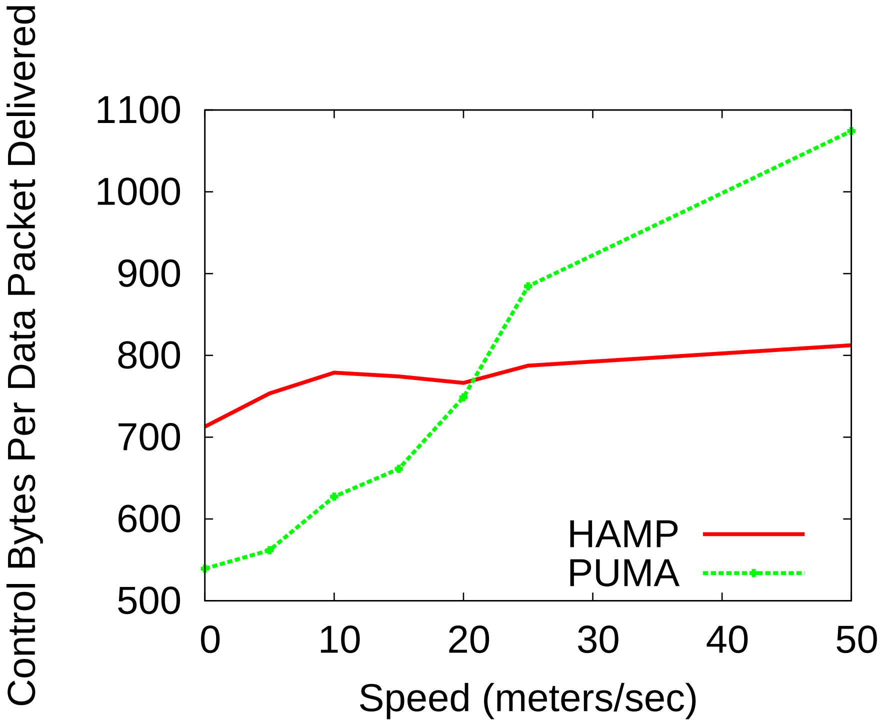


Figure 4.8: Control Overhead For Random-Street-Random Scenario

reminding that the group core in PUMA and HAMP is identified by the highest id or the first group member to join the network and does not change unless network partitions occur.

4.6.1 Packet Delivery Ratio

The results exhibited in Figure 4.9 show that HAMP and PAMPA continue to offer high delivery ratios even when the core is placed in an unstable region. However, PUMA, despite still showing sensitivity to nodes mobility, manages to deliver more packets than in scenarios with a stable core. This is explained by PUMA's multicast announcements periodically generated by the core node that make a significant part of the nodes in an unstable zone join the mesh. This fact, as will be shown when analyzing the overhead metric, comes at the cost of sacrificing the efficiency of the PUMA protocol.

Table 4.6: Control Bytes Sent per Data Packet Delivered for Random-Street-Random Scenario

		Speed						
Nodes		0 m/s	5 m/s	10 m/s	15 m/s	20 m/s	25 m/s	50m/s
2	HAMP	6865	7464	7505	7578	7670	7753	8036
	PUMA	5149	5543	6549	6923	8076	9121	10349
5	HAMP	2814	3047	3078	3051	3097	3162	3231
	PUMA	2122	2289	2730	2892	3459	4090	5146
10	HAMP	1422	1519	1553	1537	1538	1582	1640
	PUMA	1080	1119	1295	1333	1516	1782	2143
20	HAMP	712	753	778	774	766	787	812
	PUMA	539	561	627	661	748	884	1074
40	HAMP	352	386	386	389	388	395	403
	PUMA	258	282	297	329	349	398	519

4.6.2 Number of Total Bytes Transmitted per Data Packet Delivered

Figure 4.10 shows that for higher speeds (over 10 m/s), PUMA is the most expensive protocol of the three. This is explained by a high number of nodes close to the unstable core that form a mesh and a less efficient routing technique used by PUMA. However, as was shown by delivery ratios for these tests in Table 4.7, the reliance of PUMA on routing information that quickly becomes stale, leads to packet losses even at the cost of the high overhead.

Similarly, HAMP induces slightly higher total overhead than PAMPA due to its control overhead and constant attempts to perform mesh-based routing in less dynamic parts of the network.

Table 4.7: Delivery Ratio for a Scenario with Unstable Core

		Speed						
Nodes		0 m/s	5 m/s	10 m/s	15 m/s	20 m/s	25 m/s	50m/s
2	HAMP	0.9780	0.9782	0.9728	0.9748	0.9736	0.9748	0.9774
	PAMPA	1.0000	0.9976	0.9982	0.9964	0.9936	0.9996	0.9994
	PUMA	0.9786	0.9398	0.8446	0.8274	0.8280	0.8048	0.7632
5	HAMP	0.9192	0.9655	0.9594	0.9642	0.9623	0.9475	0.9585
	PAMPA	0.9999	0.9982	0.9988	0.9930	0.9850	0.9989	0.9986
	PUMA	0.9188	0.8735	0.8118	0.8047	0.7988	0.7808	0.7421
10	HAMP	0.9520	0.9743	0.9648	0.9709	0.9701	0.9623	0.9695
	PAMPA	0.9999	0.9985	0.9991	0.9962	0.9864	0.9986	0.9983
	PUMA	0.9231	0.8996	0.8524	0.8482	0.8500	0.8333	0.8106
20	HAMP	0.9560	0.9807	0.9762	0.9789	0.9738	0.9757	0.9705
	PAMPA	1.0000	0.9981	0.9988	0.9942	0.9843	0.9986	0.9983
	PUMA	0.9294	0.9078	0.8816	0.8739	0.8583	0.8609	0.8250
40	HAMP	0.9502	0.9777	0.9772	0.9727	0.9804	0.9736	0.9728
	PAMPA	0.9950	0.9984	0.9966	0.9938	0.9823	0.9985	0.9980
	PUMA	0.9145	0.9214	0.8914	0.8952	0.8832	0.8640	0.8388

Table 4.8: Total Bytes Sent per Data Packet Delivered for a Scenario with Unstable Core

		Speed						
Nodes		0 m/s	5 m/s	10 m/s	15 m/s	20 m/s	25 m/s	50m/s
2	HAMP	79377	140471	155496	161113	163213	157540	167837
	PAMPA	231833	210597	209805	204844	207784	207343	193012
	PUMA	65325	63132	81292	87156	85148	84844	85931
5	HAMP	55027	71564	75011	77846	78604	77237	78936
	PAMPA	92807	84104	83888	82201	83721	82904	77173
	PUMA	56778	58298	63713	69819	65254	67299	70098
10	HAMP	30621	36550	38407	39327	39606	39526	40102
	PAMPA	46385	42018	41927	40968	41809	41452	38602
	PUMA	35453	34065	34847	38964	39763	39189	39966
20	HAMP	15757	18510	19464	19916	20013	19931	20226
	PAMPA	23187	21017	20963	20521	20943	20721	19297
	PUMA	20103	19421	20527	22069	21935	22087	22828
40	HAMP	8997	9901	10293	10396	10445	10506	10484
	PAMPA	11649	10504	10505	10265	10491	10361	9650
	PUMA	12953	12512	13045	13663	13706	13874	14005

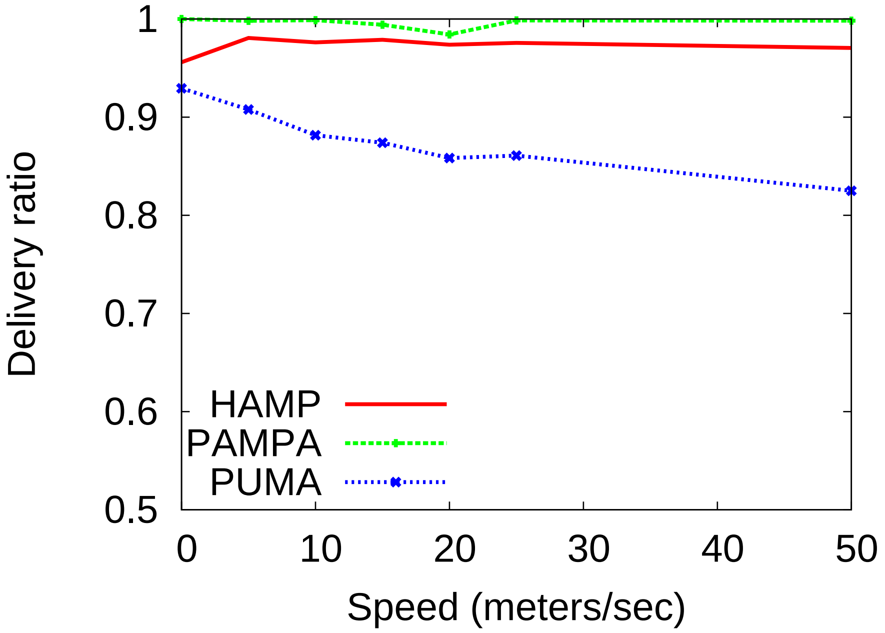


Figure 4.9: Delivery Ratios For a Scenario with Unstable Core

4.6.3 Control Overhead

Figure 4.11 show that the control overhead of the evaluated protocols is similar to the previous scenario. Both in PUMA and in HAMP, constant changes in the forwarding mesh make mesh members generate more multicast announcements. However, as the delivery ratio of the PUMA protocol deteriorates, its control overhead per every packet delivered becomes higher while control overhead induced by HAMP only suffers a significant increase initially, when mobility is first detected, and does not increase much as the node speed increases further.

4.7 Random Waypoint Scenario

In the previous tests, we showed the advantage of using the HAMP protocol in the networks with heterogeneous mobility patterns. However, HAMP also aims at achieving good performance in other scenarios, like, for instance, scenarios where all nodes are mobile.

To assess the performance of HAMP in such scenarios, we present the simulation results for

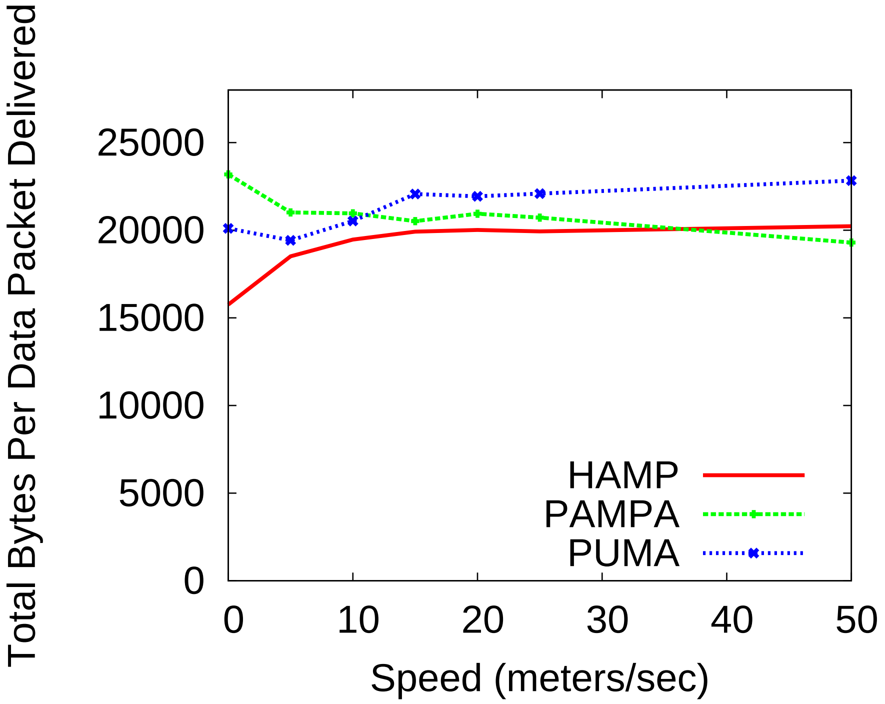


Figure 4.10: Total Overhead For a Scenario with Unstable Core

the 3 protocols in a scenario where all the nodes move in a random way. In these tests, 110 nodes move according to a Random Waypoint model in the area 300 x 300 square meters.

4.7.1 Packet Delivery Ratio

The packet delivery ratios for this scenario are presented in Table 4.10.

As can be observed, the delivery ratios for the HAMP operation are, as expected, close to those for PAMPA. Additionally, the delivery ratios achieved by PUMA are much higher than in previous tests. As all the nodes in the network are constantly moving and, as a result, much more nodes make part of the mesh, delivery ratios improve.

4.7.2 Number of Total Bytes Transmitted per Data Packet Delivered

The overall overhead of the PUMA protocol is the highest due the non-optimal routing technique used by the protocol within the mesh. As a large part of the nodes in the network belong to the mesh, the overhead induced by PUMA rises significantly. In this scenario, PAMPA

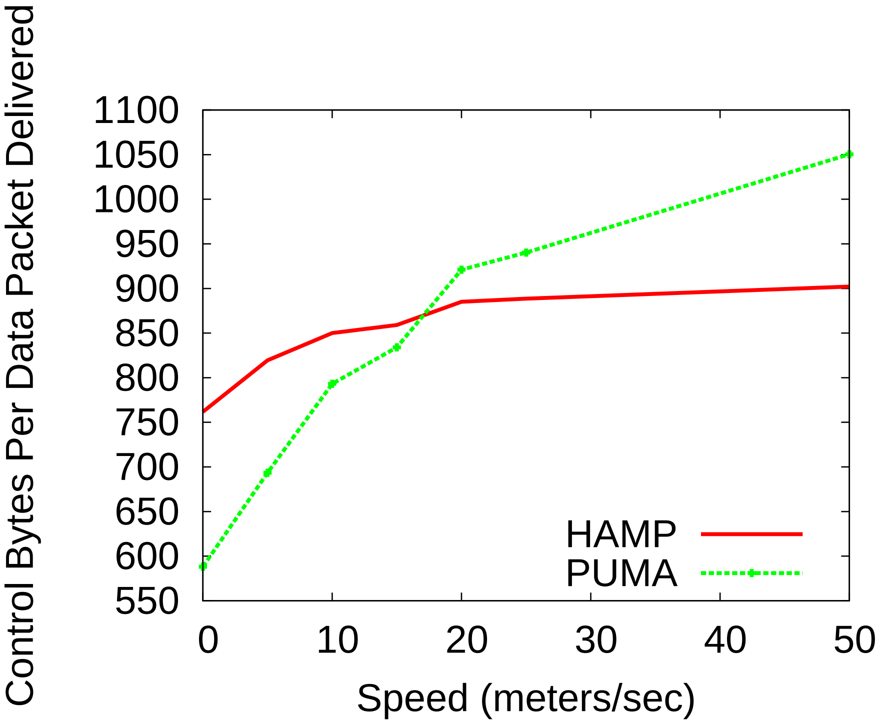


Figure 4.11: Control Overhead For a Scenario with Unstable Core

is the most efficient protocol due to its optimized message dissemination technique. HAMP, being a combination of PUMA and PAMPA, demonstrates higher overhead than PAMPA due to its control messages, flooding application in roughly entire network and additional attempts to construct a mesh and perform mesh-based forwarding along it.

The detailed evaluation results of total overhead for the protocols in this scenario can be found in Table 4.11.

4.7.3 Control Overhead

As shown in Figure 4.14, the additional control overhead of HAMP is about 30% when compared to PUMA. However, when considering the total overhead for each packet delivered, this value is not significant (only about 4% for the scenario with 20 group members). Additionally, as shown by a total overhead metric, in spite of higher control overhead induced by HAMP, the protocol is overall less expensive than the pure structured approach in this scenario.

The control overhead for other group sizes can be found in Table 4.12.

Table 4.9: Control Bytes Sent per Data Packet Delivered for a Scenario with Unstable Core

		Speed						
Nodes		0 m/s	5 m/s	10 m/s	15 m/s	20 m/s	25 m/s	50m/s
2	HAMP	6895	7629	7855	7935	8153	8212	8537
	PUMA	5149	5987	7579	8345	8620	8816	10348
5	HAMP	3048	3340	3446	3468	3551	3581	3661
	PUMA	2272	2931	3491	3878	4009	4185	4833
10	HAMP	1500	1648	1713	1723	1756	1777	1839
	PUMA	1188	1413	1672	1806	1880	1980	2193
20	HAMP	761	819	849	859	885	888	902
	PUMA	588	693	793	833	921	940	1050
40	HAMP	395	410	421	433	437	445	455
	PUMA	300	332	369	392	413	454	495

4.8 Discussion

The evaluation results show a significant advantage of using a dynamically adaptable routing technique that combines structured and unstructured forwarding to perform multicast dissemination. In all tests, the HAMP protocol demonstrated very high delivery ratios. In fact, the robustness of HAMP can be compared to that of flooding techniques. On the other hand, the total cost of the protocol is comparable to structured mesh solutions. Thus, the protocol manages to incorporate the advantages of two distinct routing approaches offering a very favorable trade-off between the efficiency and reliability.

The evaluation results demonstrated that the overhead induced by HAMP does not depend on the node speed and will only depend on the area of unstable zones where the mesh is formed. This means that, in worst case, HAMP performs packet forwarding by flooding through the entire network, thus, benefiting from the robustness of such an expensive approach. On the other

Table 4.10: Delivery Ratio for Random Waypoint Scenario

		Speed						
Nodes		0 m/s	5 m/s	10 m/s	15 m/s	20 m/s	25 m/s	50m/s
2	HAMP	0.9814	0.9832	0.9824	0.9862	0.9912	0.9876	0.9742
	PAMPA	0.9866	0.9826	0.9860	0.9886	0.9938	0.9888	0.9768
	PUMA	0.9436	0.9126	0.8744	0.8952	0.8842	0.8672	0.8550
5	HAMP	0.9662	0.9736	0.9716	0.9833	0.9910	0.9769	0.9697
	PAMPA	0.9750	0.9781	0.9825	0.9803	0.9889	0.9771	0.9726
	PUMA	0.9184	0.8812	0.8576	0.8757	0.8680	0.8541	0.8290
10	HAMP	0.9620	0.9732	0.9803	0.9800	0.9849	0.9729	0.9705
	PAMPA	0.9728	0.9761	0.9826	0.9826	0.9852	0.9730	0.9731
	PUMA	0.9226	0.9113	0.8929	0.9128	0.8929	0.8687	0.8668
20	HAMP	0.9590	0.9707	0.9771	0.9778	0.9838	0.9687	0.9745
	PAMPA	0.9674	0.9734	0.9807	0.9805	0.9823	0.9704	0.9720
	PUMA	0.9335	0.9320	0.9294	0.9317	0.9358	0.9146	0.9099
40	HAMP	0.9602	0.9707	0.9792	0.9839	0.9830	0.9704	0.9781
	PAMPA	0.9678	0.9726	0.9788	0.9785	0.9810	0.9685	0.9726
	PUMA	0.9486	0.9505	0.9538	0.9594	0.9605	0.9399	0.9432

Table 4.11: Total Bytes Sent per Data Packet Delivered for Random Waypoint Scenario

		Speed						
Nodes		0 m/s	5 m/s	10 m/s	15 m/s	20 m/s	25 m/s	50m/s
2	HAMP	97466	134826	146326	148675	150840	151082	155886
	PAMPA	118383	127280	130419	132531	134329	134599	140663
	PUMA	42356	39846	57285	52114	52410	44085	47255
5	HAMP	54821	61090	62417	62855	63193	62981	63969
	PAMPA	47917	51141	52359	53445	53905	54457	56496
	PUMA	45294	46109	49910	47999	48287	45939	48457
10	HAMP	30277	31453	31922	31819	31980	32147	32282
	PAMPA	23997	25612	26179	26657	27051	27340	28225
	PUMA	31196	31323	33192	32442	32377	32248	32088
20	HAMP	15837	16184	16246	16218	16201	16339	16376
	PAMPA	12065	12840	13113	13355	13566	13706	14127
	PUMA	19819	19912	20607	20136	20147	20145	20261
40	HAMP	8069	8247	8288	8244	8252	8284	8314
	PAMPA	6031	6426	6569	6690	6791	6865	7057
	PUMA	11864	11848	12027	11899	11980	11983	12092

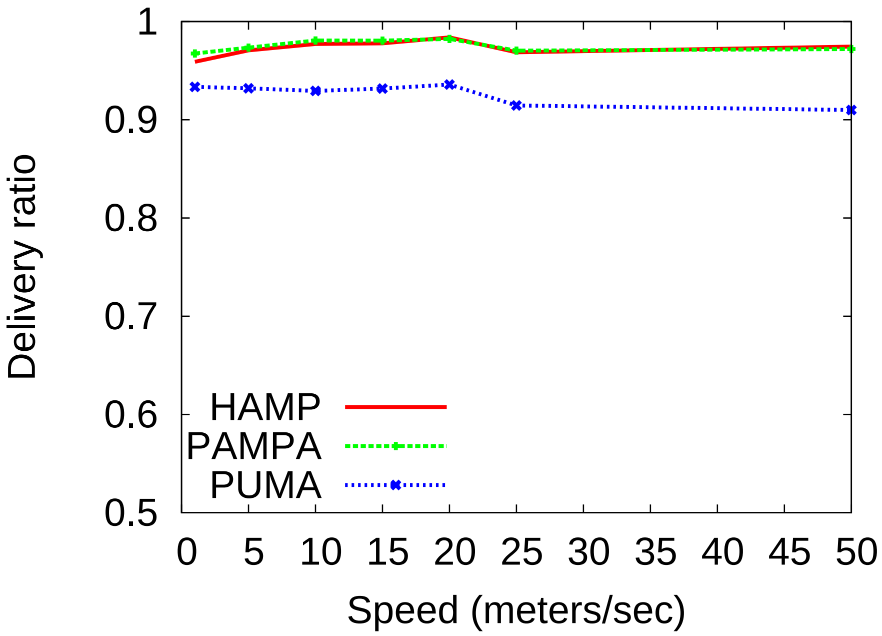


Figure 4.12: Delivery Ratios For Random Waypoint Scenario

hand, as we saw, mesh protocols, in spite of being less costly, do not manage to demonstrate high robustness under heterogeneous mobility conditions.

One of the main contributions of the HAMP protocol is the definition of stability that is used to detect mobility in MANET environments. The technique proved to be able to identify unstable zones in the network and calculate a diameter sufficient to cross the unstable zones by scoped flooding.

However, we observed that the instability condition may also generate false positives. For instance, we observed the following phenomenon in static scenarios: in dense parts of the mesh, the random delay in rebroadcasting multicast announcements makes the mesh constantly change; in these (typically small) areas, HAMP infers instability and forwards multicast traffic by a scoped flooding scheme.

Interestingly, in these corner cases, the false instability detection is actually beneficial to the HAMP operation, as the optimized flooding scheme is more efficient than the mesh-based routing in very dense regions. With a small group size and, as a result, a thin mesh, this artifact

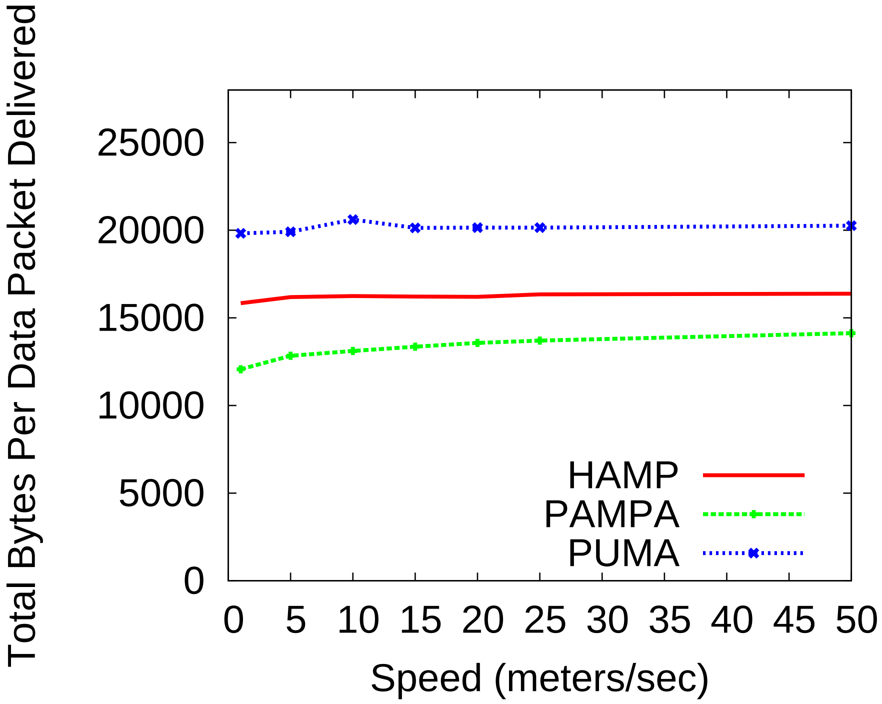


Figure 4.13: Total Overhead For Random Waypoint Scenario

occurs less. But, as a group size grows, it becomes more visible. This actually explains a smaller total overhead of HAMP per data packet delivered with zero mobility, as shown in Figure 4.7.

On the other hand, this phenomenon shows that the randomness present in the PUMA protocol changes slightly the expected outcome of the stability condition as it is defined in HAMP. This artifact can be mitigated by increasing a k parameter in order the node to be able to ignore temporary mesh changes. On the other hand, a higher value of k will delay the identification of actual mesh partitions and will lead to higher packet loss ratios.

As we stated initially, the hybrid approach represents a trade-off between efficiency and reliability. If group members are primarily located in unstable parts of the network or if the entire network is highly mobile, the structured protocols tend to generate a lot of control traffic and still do not manage to keep high delivery ratios. It was shown in the scenario with an unstable group core that, under these conditions, the protocol will perform flooding in order to reach highly mobile group members. This also will lead to higher overheads; however, the delivery ratios will not be affected by the mobility.

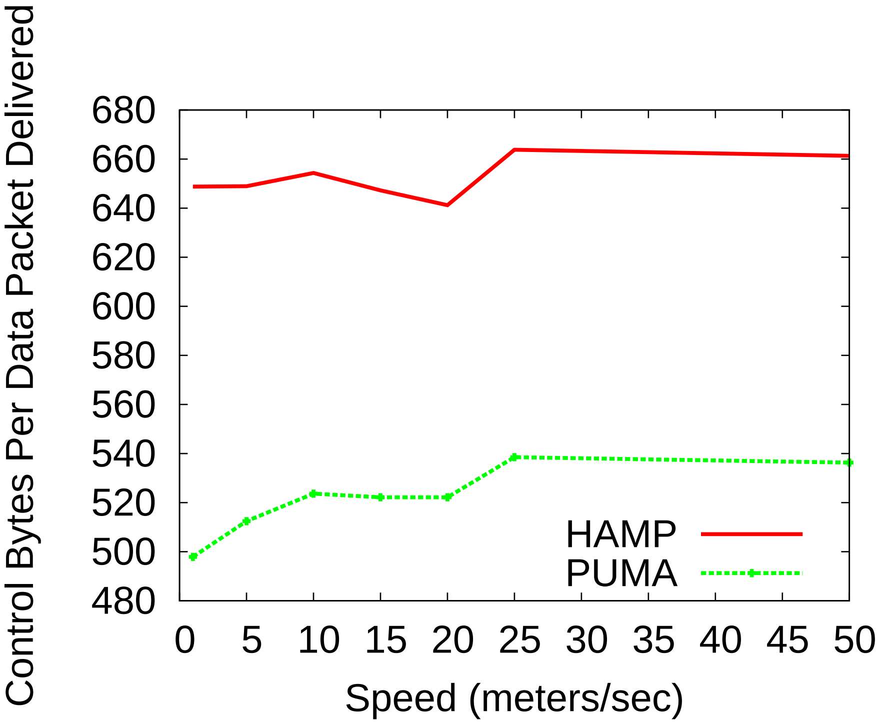


Figure 4.14: Control Overhead For Random Waypoint Scenario

Summary

In this Chapter, we have experimentally evaluated the operation of HAMP using simulations. We have compared its performance against solutions designed for homogeneous networks, such as PAMPA and PUMA.

Overall, HAMP proved to be a robust and efficient routing approach for networks with heterogeneous mobility patterns that, when needed, sacrifices efficiency in favor of greater reliability.

Table 4.12: Control Bytes Sent per Data Packet Delivered for Random Waypoint Scenario

		Speed						
Nodes		0 m/s	5 m/s	10 m/s	15 m/s	20 m/s	25 m/s	50m/s
2	HAMP	5766	6113	6203	6235	6266	6389	6476
	PUMA	4380	4610	5107	4934	5026	5205	5296
5	HAMP	2483	2565	2558	2578	2537	2601	2605
	PUMA	1962	2119	2243	2202	2248	2299	2442
10	HAMP	1278	1305	1281	1298	1291	1305	1311
	PUMA	1014	1056	1095	1084	1108	1154	1170
20	HAMP	648	648	654	647	641	663	661
	PUMA	497	512	523	522	522	538	536
40	HAMP	325	324	320	321	322	325	330
	PUMA	233	236	239	237	240	245	248

5

Conclusions and Future Work

5.1 Conclusions

Many realistic ad hoc networks of the future will exhibit heterogeneous mobility patterns. Thus, protocols designed for MANET environments, including multicast protocols, should be able to adapt to this sort of heterogeneity. This thesis has addressed the problem of building such protocols.

A key aspect in the design of any multicast protocol is to reduce the amount of control traffic and redundant messages transmitted in the network without sacrificing reliability. One of the main approaches to do so consists in building some form of overlay to support the multicast operation. Unfortunately, the structure of such overlay may become unstable or even break under high mobility. On the other hand, unstructured approaches, such as optimized forms of flooding, are very resilient to mobility but inefficient in terms of network usage.

This thesis proposed HAMP – Heterogeneity-Aware Multicast Protocol, a novel approach that combines structured and unstructured approaches to achieve a very favorable tradeoff between delivery rate and message cost in networks with heterogeneous mobility patterns. This is possible because unstable regions are dynamically identified and crossed using a scoped flooding that remains confined to those regions.

HAMP was implemented as a combination of PUMA – Protocol for Unified Multicasting through Announcements, a mesh-based multicast protocol, and PAMPA – Power-Aware Message Propagation Algorithm, an optimized flooding protocol to cross unstable regions. However, the key aspects of the solution can also be applied to combine other structured and unstructured approaches.

5.2 Future Work

As future work we would like to extend the approach to accommodate other multicast strategies for both stable and unstable regions.

Also, the ability to identify which nodes are stable is key to our approach. The favorable results presented in this work demonstrate the potential of a system that is able to identify such nodes. Additional research on how to characterize and identify stability in MANETs can bring further advances to design of network protocols.

Finally, the proposed mechanism of combining structured and unstructured routing is not limited to multicast protocols. We would also like to extend the mechanism to other protocols, including unicast and broadcast protocols.

Bibliography

Ballardie, T., P. Francis, & J. Crowcroft (1993). Core Based Trees (CBT). *SIGCOMM '93: Conference on Communications Architectures, Protocols and Applications 23*(4), 85–95.

Das, S. K., B. S. B. S. Manoj, & C. S. R. Murthy (2002). A Dynamic Core Based Multicast Routing Protocol for Ad Hoc Wireless Networks. In *MobiHoc '02: Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing*, New York, NY, USA, pp. 24–35. ACM.

Denysyuk, O., J. Mocito, & L. Rodrigues (2009). HAMP – Protocolo de Difusão em Grupo Para Redes Com Padrões de Mobilidade Heterogéneos. In *Actas da 9ª Conferência sobre Redes de Computadores*, Oeiras, Portugal.

Dolev, S., E. Schiller, & J. L. Welch (2006). Random Walk for Self-Stabilizing Group Communication in Ad Hoc Networks. *IEEE Transactions on Mobile Computing 5*(7), 893–905.

Drabkin, V., R. Friedman, G. Kliot, & M. Segal (2007). RAPID: Reliable Probabilistic Dissemination in Wireless Ad-Hoc Networks. In *Proceedings of the 26th IEEE International Symposium on Reliable Distributed Systems.*, Beijing, China, pp. 13–22. IEEE.

Farhan, K. A. (2008). Network Sender Multicast Routing Protocol. In *ICN '08: Proceedings of the Seventh International Conference on Networking (icn 2008)*, Washington, DC, USA, pp. 60–65. IEEE.

Garbinato, B., A. Holzer, & F. Vessaz (2008). Six-Shot Broadcast: A Context-Aware Algorithm for Efficient Message Diffusion in MANETs. In *OTM '08: Proceedings of the OTM 2008 Confederated International Conferences, CoopIS, DOA, GADA, IS, and ODBASE 2008. Part I on On the Move to Meaningful Internet Systems:*, Berlin, Heidelberg, pp. 625–638. Springer-Verlag.

Haas, Z. J., J. Y. Halpern, & L. Li (2006). Gossip-Based Ad Hoc Routing. *IEEE/ACM Transactions on Networking 14*(3), 479–491.

Jaikaeo, C. & C.-C. Shen (2002). Adaptive Backbone-based Multicast for Ad Hoc Networks. In

Proceedings of the IEEE International Conference on Communications (ICC), pp. 3149–3155. IEEE.

Jetcheva, J. G. & D. B. Johnson (2001). Adaptive Demand-driven Multicast Routing in Multi-hop Wireless Ad Hoc Networks. In *MobiHoc '01: Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking & computing*, New York, NY, USA, pp. 33–44. ACM.

Ji, L. & M. S. Corson (2001). Differential Destination Multicast: A MANET Multicast Routing Protocol for Small Groups. In *INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies*, Volume 2, pp. 1192–1201 vol.2. IEEE.

Kunz, T. & E. Cheng (2002). On-Demand Multicasting in Ad-Hoc Networks: Comparing AODV and ODMRP. In *Proceedings of the International Conference on Distributed Computing Systems*, pp. 453–454. IEEE.

Lee, S.-J., W. Su, & M. Gerla (1999). On-Demand Multicast Routing Protocol. In *WCNC: Wireless Communications and Networking Conference*, Volume 3, pp. 1298–1302. IEEE.

Miranda, H., S. Leggio, L. Rodrigues, & K. Raatikainen (2006). Power-Aware Broadcasting Algorithm. In *Proceedings of the 17th Annual IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, Helsinki, Finland, pp. 1–5. IEEE.

Ni, S.-Y., Y.-C. Tseng, Y.-S. Chen, & J.-P. Sheu (1999). The Broadcast Storm Problem in a Mobile Ad Hoc Network. In *MobiCom '99: Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, Seattle, Washington, United States, pp. 151–162. ACM.

Obraczka, K., K. Viswanath, & G. Tsudik (2001). Flooding for Reliable Multicast in Multi-Hop Ad Hoc Networks. *Wireless Networks 7*(6), 627–634.

Royer, E. M. & C. E. Perkins (1999). Multicast Operation of the Ad Hoc On-demand Distance Vector Routing Protocol. In *MobiCom '99: Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, New York, NY, USA, pp. 207–218. ACM.

Vaishampayan, R. & J. J. Garcia-Luna-Aceves (2004). Efficient and Robust Multicast Routing in Mobile Ad Hoc Networks. In *Proceedings of the International Conference on Mobile Ad-*

hoc and Sensor Systems, pp. 304–313. IEEE.

Voulgaris, S., E. Riviere, A.-M. Kermarrec, & M. van Steen (2006). Sub-2-Sub: Self-Organizing Content-Based Publish and Subscribe for Dynamic and Large Scale Collaborative Networks. In *International workshop on Peer-To-Peer Systems*, Santa Barbara, CA, USA. ACM.

Wu, C. W. & Y. C. Tay (1999). AMRIS: A Multicast Protocol for Ad Hoc Wireless Networks. In *Military Communications Conference Proceedings*, pp. 25–29. IEEE.

Xie, J., R. R. Talpade, A. Mcauley, & M. Liu (2002). AMRoute: Ad Hoc Multicast Routing Protocol. *Mobile Networks and Applications* *7*(6), 429–439.