Group Communication in Mobile Ad Hoc Networks

Oksana Denysyuk
oksana.denysyuk@ist.utl.pt

Instituto Superior Técnico

(Advisor: Professor Luís Rodrigues)

Abstract. This report addresses the problem of supporting multicast in mobile ad hoc networks (MANETs). Multicast is an important building block for many applications in MANETs, including data dissemination, service discovery, publish-subscribe, among others. It has been therefore widely studied and many solutions can be found in the literature. However, most solutions are tailored to a specific type of mobility pattern and are unable to excel in face of heterogeneous mobility conditions. In this report we survey and discuss different multicast protocols for MANETs and sketch a novel protocol for MANETs with heterogeneous mobility patterns that combines the efficiency of tree-based approaches and the robustness of flooding-based schemes.

1 Introduction

A *mobile ad hoc network* (MANET) 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 operation, search and rescue, military operations, ad hoc gaming, etc. Many of these application 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 group communication support at the middleware level. Multicast, the ability to send a message to a group of processes, is a central component of any group communication service. This report focuses on multicast protocols for mobile ad hoc environments.

The multicast problem has been widely studied, both for wired and wireless networks. 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.

Therefore, multicast protocols designed for wired environments do not perform well in MANETs. Even the most dynamic wired environments, such as peer-to-peer (P2P) networks, use multicast protocols that are not adequate to MANETs, for instance, they may rely on the existence of a highly available rendez-vous point. Therefore, a significant amount of work has been performed in designing multicast protocols for the MANET environment. These solutions 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.

As it will be seen later in the report, 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 routes among nodes. Structured approaches create some form of a multicast-tree. They trade the cost of building the tree for a more efficient message dissemination procedure in stable conditions. They are therefore well adapted to scenarios with low or sporadic mobility.

It is our belief that many of the 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. The goal of our research is to analyze the existing solutions for multicast in MANETs and to explore the viability of providing an efficient yet robust multicast protocol for mobile ad hoc networks that combines an efficient structured tree-based approach and a robust network flooding.

The rest of the report is organized as follows. Section 2 briefly summarizes the goals and expected results of our work. In Section 3, we make a brief survey of the existing broadcast and multicast protocols, discuss general design alternatives and position the multicast in generic group communication services. Based on this discussion we also explore the need for new protocols that provide high delivery guarantees with low overhead. In Section 4, we provide a sketch of a hybrid multicast protocol that combines the efficiency of a structured tree-based solution with the robustness of flooding techniques. Section 5 describes how we plan to evaluate our proposal. Finally, Section 6 presents the schedule of future work and Section 7 concludes the report.

2 Goals

This work addresses the problem of providing multicast support in MANETs. In particular we are interested in realistic scenarios, that exhibit heterogeneous mobility patterns, i.e., regions with low mobility and regions with high mobility.

> *Goals:* This works is focused on designing a new multicast protocol for MANETs that operates efficiently in networks that have a mix of high mobility and low mobility regions.

As it will become clear later in the text, our protocol will embed a combination of flooding-based and tree-based mechanisms. In order to evaluate the proposed solution we will resource to simulation. In particular, we plan to use the widely adopted NS-2 simulator. An extensive experimental evaluation of the protocol in different scenarios will be performed using this tool. Therefore, the project will produce the following expected results.

> *Expected results:* The work will produce the i) specification of the protocol; ii) an implementation of the protocol for the NS-2 simulator and, iii) an extensive experimental evaluation of the protocol using simulations.

3 Multicast in MANETs

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 identity 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 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 section we survey different techniques to implement multicast in MANETs. We will be mainly concerned with protocols that provide a best-effort service but, for self-containment, we will briefly address mechanisms to provide stronger reliability guarantees.

3.1 Broadcast

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.

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 (to enhance performance, nodes are required to keep the identity of previously flooded message for some amount of time), in which case the node rebroadcasts the packet. 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 *time to live* (TTL) 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*[20], 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*[19]. 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.

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[11]. A source sends a message with probability 1. When a node first receives a packet, with probability p it broadcasts it to the neighbors and with probability 1 - p it discards the message. The parameter p is a previously defined number.

The 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.

Counter Based Methods One way to estimate the density dynamically is what is called the *counter based scheme*. 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. When a timer expires and a predefined *counter threshold* has not 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[19].

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[18] 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 Global Positioning System (GPS). Every node attaches 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[9]. 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 an hexagon centered at the source, to propagate a message in all geographical directions.

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 or two hop neighbors (and, in some cases, about the state of these neighbors, for instance, available battery).

An example of a neighbor knowledge protocol is RAPID[7] 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 with 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.

3.2 Multicast

Multicast differs from broadcast because 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[1] 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[22], ODMRP[17], DCMP[5], NSMRP[8] 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[22] and AMRIS[23] are examples of tree constructing 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[17] and DCMP[5] 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[14] is an example.

– *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[24] is an example of such a protocol.

In the next paragraphs we give some significant examples that illustrate the several design choices above.

MAODV The Multicast On Demand Distance Vector Routing Protocol[22] 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 is characterized by frequent node failures and therefore demand more robustness of the protocol. That may be achieved using redundancy of data forwarding routes.

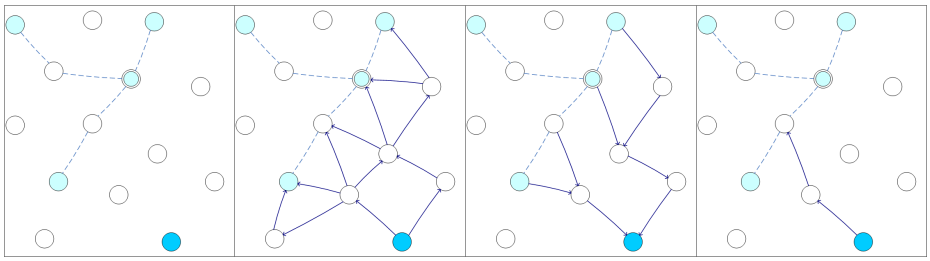
Each node in the network maintains three routing tables. The first one, simply called Routing Table, records the next hop for unicast routes to other nodes. The second, called Multicast Routing Table, contains entries for the multicast routing groups of which the node is a router. The third, named the 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 also sets the J-flag of the RREQ. 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.



(a) Initial Tree (b) RREQ (c) RREP (d) MACT

Fig. 1: MAODV Join Procedure

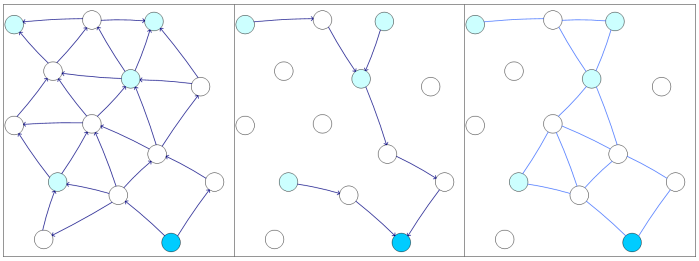
A Group Hello Message is constantly broadcast by the multicast group leader over the whole network to announce its ID and the group sequence number. Upon receiving this message, nodes update their multicast route tables with the group ID, group leader's ID and the sequence number. The Group Hello Message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

ODMRP The On Demand Multicast Routing Protocol[17] 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 neighbors and only the members of the forwarding group will rebroadcast 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.



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

Fig. 2: ODMRP Operation

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.

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.

ADMR The periodic signaling used by some structured protocols may substantially limit the benefits of the protocol's on-demand operation. The Adaptive Demand-Driven Multicast Routing protocol (ADMR)[13] 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 and keep-alives 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 hasn't 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.

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. This problem may be tackled by creating an overlay structure that only involves group members, and relying on the underlying unicast protocol to deal with network topology changes. AMRoute[24] 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 descover other members of the group. When a member node receives a JOIN_REQ from a core for 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 it 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.

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. But 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 Adaptive Backbone-Based Multicast (ADB) described in[12]. 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 the 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 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.

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[6]. 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.

3.3 Reliable Multicast

Reliable communication is an important requirement for many multicast applications and, in MANETs, it becomes a very challenging research problem due to frequent packet losses and nodes mobility. Most existing multicast protocols for MANETs provide no reliability guarantees at all. However, there are multiple recovery techniques that can be combined with best-effort protocols to provide reliability. Examples of such techniques are: Automatic Retransmission Request Based, Gossip Based and Forward Error Correction Based schemes.

Automatic Retransmission Request Based This technique consists in having the sender detect a packet loss, either by a missing acknowledgment message (ACK) or via a negative acknowledgment (NACK). In an ACK-based approach,

the receivers acknowledge every received packet. A sender is responsible to detect message losses based on the ACKs it has not received. In a NACK-based scheme, the receivers are responsible for recognizing missing packets and notify the sender with NACKs for retransmission. Automatic Retransmission Request Based mechanisms are used in the following protocols: Reliable Multicast Algorithm (RMA)[10] that applies an ACK-based approach and RALM[15] that uses NACKs to detect message losses.

Gossip Based Mechanisms This technique requires group members to periodically retransmit information about the most recently received messages to a random set of neighbors. Alternatively, nodes may propagate information about missing packets and the nodes that have those messages may retransmit them to the gossiping node.

Anonymous Gossip (AG)[3] implements gossip-based recovery on top of a multicast operation. A group member periodically transmits a gossip request message about missing and successfully received packets to a random neighbor. Upon receiving a gossip request, a non-group-member forwards the packet to one of its neighbors. A group member neighbor will respond to the gossip request with a certain probability. Otherwise it forwards the packet again. The propagation of this gossip message ends when some node replies to it or the lifetime of the message expires. The accepting node then unicasts a gossip reply to the initiator of this gossip request and the nodes will exchange the missing packets. In[3], there was suggested a use of this technique on top of MAODV, where the tree members exchange gossip messages. Thus, the gossip messages are only propagated along the multicast tree.

Forward Error Correction Schemes This technique consists in retransmiting redundant data with the original data transmission. Thus, when message losses and errors occur, the receiver may reconstruct the original data by using redundant information contained in other received packets. This technique is used by RMDP[21], that in addition applies a NACK-based mechanism to recover packets that cannot be reconstructed.

3.4 Group Communication

Many best-effort multicast services, namely for applications such as streaming, do not explicitly keep track of the group membership. In fact, in many of these applications, none of the participants has an up-to-date view of who are the group members. If multicast and group join operations are executed concurrently, a message may be delivered to the new group members based on practical operational factors and not as a result of attempting to ensure some precise semantics.

However, when the group members need to coordinate, for instance to reach agreement on some specific action, it may be important to give to each participant updated information about the group membership and also to define precisely the semantics of reliable multicast in face of such dynamic membership. A

system that provides such services is commonly called a *group communication system*[4].

Group communication systems traditionally provide a *membership service* which maintains the information about currently connected active members of a group, called the group *view*. When a list of members changes, the membership service reports the change to the members by *installing* a new view.

One important aspect of a group membership service is how it reacts to network partitions. A *primary-partition* membership service only installs views on one network partition, which is called primary partition. Processes in the parts of the network which are not connected to the primary partition are considered faulty. On the other hand, *partitionable* group membership service allows multiple network partitions to run the service simultaneously. If a group splits due to network partitions, all its parts continue working. If the network becomes connected again, the parts of the group merge.

Given that the membership is dynamic, it becomes non-trivial to define precise semantics for multicast reliability. The most widely accepted is called *view synchrony*[2] and is defined as follows: two processes that participate in two consecutive views V' and V" deliver the same set of messages in the previous view V'.

3.5 Message Ordering

Besides ensuring that messages are reliably delivered, it is also possible to run protocols that enforce some message ordering guarantees. By providing such service in the communication or middleware layers, one can greatly simplify the application design. The most common order guarantees are FIFO, Causal and Total orderings, as described below.

- *FIFO* (First-In-First-Out) delivery requires that all messages sent by the same source are delivered to all the receivers in the order they were sent.
- *Causal* delivery extends the FIFO semantics by guaranteeing that if a node sends some message m after receiving m' then all the nodes will deliver m after delivering m'. The rational behind this ordering scheme is that if a message m is a response to m', then all the receivers of m will see the response after the original message.
- *Total* ordering guarantee means that all messages are delivered in the same order at all the correct processes that deliver them. There exists a stronger variant of this guarantee, called *Uniform Total* ordering that requires the messages be delivered in the same order at *all* the processes that deliver them. It means that even if a process delivers a message m before m' and crashes immediately after that, all the correct processes in the group must deliver m before m'.

3.6 Discussion and Comparison

All the multicast algorithms presented before have advantages and drawbacks.

Flooding is not resource efficient in the general case, because 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 a 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 are 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 then 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 [16] 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 phenomena. In static networks, ODMRP introduces high overheads due to constant broadcasts not only by route refreshing procedure, 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 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.

Contrarily to tree 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, but, after a careful ADB analysis, 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.

4 Multicast in MANETs with Heterogeneous Mobility Patterns

As discussed in the previous section, multicast protocols are often optimized for some mobility pattern. In ad hoc networks that exhibit some stability, it is worth to build a tree to support the multicast. Tree-based solutions are more efficient because they minimize the number of nodes that need to forward the multicast message. However, in very dynamic systems, the tree is unstable, as node movement breaks one or more tree branches. This forces the algorithm to rebuild the tree, which is an expensive operation that often requires some form of flooding. Since flooding is required to build and repair a tree, in highly dynamic environments it may be worth to just rely on some form of flooding to support the multicast operation.

The problem addressed in this project is to design a multicast protocol that can exhibit good performance in heterogeneous environments, that have a combination of stable and highly dynamic regions. As we have stated in the introduction, it is our belief that many realistic environments will have this characteristic. The rational behind our proposal is that it should be possible to use a tree-based approach in stable network regions and a flooding-based approach on dynamic regions. Both approaches may co-exist in the same network and a seamless transition from tree-based to flood-based operation should be supported by the protocol. In fact, the ad hoc domain may be composed of multiple stable and unstable regions and a multicast may commute multiple times between tree-based and flood-based forwarding.

Briefly, we sketch the main components of the proposed solution:

- A flood-based forwarding scheme, using some of the algorithms presented in the previous section, most likely PAMPA[18].
- A tree-based forwarding scheme, using some of the algorithms presented in the previous section, most likely MAODV[22].

– An algorithm that allows to commute from one forwarding scheme to the other.

The main challenge of our proposal is to design the mechanisms that allow flood-based and tree-based forwarding schemes to co-exist. In particular, the following problems have no trivial solution:

– When a node receives a multicast message from a tree branch it needs to decide if the next hop is still a stable tree-member or a node in an unstable region. In the first case, it forwards the message using the tree-based scheme and, in the second case, it needs to initiate a flooding procedure in the unstable region.
– When a node receives a multicast message from flooding it needs to decide if it propagates the flooding or just unicasts the message to a stable neighbor. The key idea is that some techniques must be devised to contain the flooding in the unstable network region and prevent it from propagating into stable areas.

Thus, while nodes inside stable/unstable regions can operate exclusively using tree-based/flood-based algorithms, nodes in the boundaries of stable and unstable zones will play a special role, by commuting between the two modes of the protocol. Naturally, the role of each node can not be defined a priory and is not immutable. As nodes move and topology changes, any node may enter or leave a boundary.

5 Evaluation

We will evaluate our protocol experimentally using the widely adopted NS-2 network simulator. We are planning to use this simulator because it already includes implementations of many unicast and multicast protocols for MANETs. It is also the simulator of choice in the most representative publications in the field and, therefore, our results will be comparable with other published results in the literature.

The evaluation of the protocol will use two main metrics: the *reliability of the protocol*, i.e., the percentage of multicast messages that are effectively delivered to their targets, and the *overhead of the protocol*, i.e., the total amount of data and control messages exchanged by the protocol. We will evaluate these metrics in different scenarios, by changing the ratio of mobile and more stable nodes, the mobility patterns of mobile nodes, and the global network topology.

In order to have comparative values, we will compare the performance of our protocol against the performance of pure flooding-based and pure tree-based protocols operating in the same scenarios. Hopefully, our protocol will exhibit a better reliability/overhead ratio.

6 Scheduling of Future Work

Future work is scheduled as follows:

- January 9 - March 29: Detailed design and implementation of the proposed architecture, including preliminary tests.
- March 30 - May 3: Perform the complete experimental evaluation of the results.
- May 4- May 23, 2009: Write a paper describing the project.
- May 24 - June 15: Finish the writing of the dissertation.
- June 15, 2009: Deliver the MSc dissertation.

7 Conclusions

In this report we surveyed the most representative multicast protocols for MANETs, focusing on the benefits and disadvantages of each of them. As a result of the discussion we sketched the basis of a multicast protocol for disseminating data in a mobile ad hoc network with heterogeneous mobility patterns, that will combine the most robust and the most efficient features of the existing solutions. Our solution will infer local network stability and employ a structured solution in more static environments; when significant mobility is observed, a flooding technique will be applied locally. In this way, the new protocol will dynamically adapt its behavior according to observed network conditions minimizing resource consumption without compromising reliability. We concluded the report with a description of the methodology to be applied for evaluating our solution.

Acknowledgments

I am very grateful to my advisor Professor Luís Rodrigues and to my colleagues José Mocito and João Leitão, for the fruitful discussions and comments during the preparation of this report.

References

1. Ballardie, T., Francis, P., Crowcroft, J.: Core based trees (cbt). SIGCOMM Comput. Commun. Rev. **23**(4), 85–95 (1993). DOI http://doi.acm.org/10.1145/167954.166246
2. Birman, K., Joseph, T.: Exploiting virtual synchrony in distributed systems. In: SOSP '87: Proceedings of the eleventh ACM Symposium on Operating systems principles, pp. 123–138. ACM, New York, NY, USA (1987). DOI http://doi.acm.org/10.1145/41457.37515
3. Chandra, R., Ramasubramanian, V., Birman, K.: Anonymous gossip: improving multicast reliability in mobile ad-hoc networks. Distributed Computing Systems, 2001. 21st International Conference on. pp. 275–283 (2001). DOI 10.1109/ICDSC.2001.918957

4. Chockler, G.V., Keidar, I., Vitenberg, R.: Group communication specifications: a comprehensive study. ACM Comput. Surv. **33**(4), 427–469 (2001). DOI http://doi.acm.org/10.1145/503112.503113
5. Das, S.K., Manoj, B.S.B.S., Murthy, C.S.R.: 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, pp. 24–35. ACM, New York, NY, USA (2002). DOI http://doi.acm.org/10.1145/513800.513804
6. Dolev, S., Schiller, E., Welch, J.L.: Random walk for self-stabilizing group communication in ad hoc networks. IEEE Transactions on Mobile Computing **5**(7), 893–905 (2006). DOI http://doi.ieeecomputersociety.org/10.1109/TMC.2006.104
7. Drabkin, V., Friedman, R., Kliot, G., Segal, M.: Rapid: Reliable probabilistic dissemination in wireless ad-hoc networks. Reliable Distributed Systems, 2007. SRDS 2007. 26th IEEE International Symposium on pp. 13–22 (2007). DOI 10.1109/SRDS.2007.9
8. Farhan, K.A.: Network sender multicast routing protocol. In: ICN '08: Proceedings of the Seventh International Conference on Networking (icn 2008), pp. 60–65. IEEE Computer Society, Washington, DC, USA (2008). DOI http://dx.doi.org/10.1109/ICN.2008.67
9. Garbinato, B., Holzer, A., Vessaz, F.: Six-shot broadcast: A context-aware algorithm for efficient message diffusion in manets. In: R. Meersman, Z. Tari (eds.) OTM Conferences (1), *Lecture Notes in Computer Science*, vol. 5331, pp. 625–638. Springer (2008). URL http://dblp.uni-trier.de/db/conf/otm/otm2008-1.html#GarbinatoHV08
10. Gopalsamy, T., Singhal, M., Panda, D., Sadayappan, P.: A reliable multicast algorithm for mobile ad hoc networks. Distributed Computing Systems, International Conference on **0**, 563 (2002). DOI http://doi.ieeecomputersociety.org/10.1109/ICDCS.2002.1022306
11. Haas, Z.J., Halpern, J.Y., Li, L.: Gossip-based ad hoc routing. IEEE/ACM Trans. Netw. **14**(3), 479–491 (2006). DOI http://dx.doi.org/10.1109/TNET.2006.876186
12. Jaikaeo, C., Shen, C.C.: Adaptive backbone-based multicast for ad hoc networks. Communications, 2002. ICC 2002. IEEE International Conference on **5**, 3149–3155 vol.5 (2002). DOI 10.1109/ICC.2002.997417
13. Jetcheva, J.G., Johnson, D.B.: 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, pp. 33–44. ACM, New York, NY, USA (2001)
14. Ji, L., Corson, M.S.: 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. Proceedings. IEEE, vol. 2, pp. 1192–1201 vol.2 (2001). URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=916314
15. K. Tang K. Obraczka, S.J.L.M.G.: Reliable adaptive lightweight multicast protocol. ICC (2004)
16. Kunz, T., Cheng, E.: On-demand multicasting in ad-hoc networks: Comparing aodv and odmrp. Distributed Computing Systems, International Conference on **0**, 453 (2002). DOI http://doi.ieeecomputersociety.org/10.1109/ICDCS.2002.1022287
17. ju Lee, S., Su, W., Gerla, M.: On-demand multicast routing protocol. Wireless Communications and Networking Conference

18. Miranda, H., Leggio, S., Rodrigues, L., Raatikainen, K.: A power-aware broadcasting algorithm. DI/FCUL TR 06–5, Department of Informatics, University of Lisbon (2006). URL http://www.di.fc.ul.pt/tech-reports/06-5.pdf
19. Ni, S.Y., Tseng, Y.C., Chen, Y.S., Sheu, J.P.: The broadcast storm problem in a mobile ad hoc network pp. 151–162 (1999). DOI http://doi.acm.org/10.1145/313451.313525
20. Obraczka, K., Viswanath, K., Tsudik, G.: Flooding for reliable multicast in multihop ad hoc networks. Wireless Networks **7**(6), 627–634 (2001)
21. Rizzo, L., Vicisano, L.: Rmdp: an fec-based reliable multicast protocol for wireless environments. SIGMOBILE Mob. Comput. Commun. Rev. **2**(2), 23–31 (1998). DOI http://doi.acm.org/10.1145/584017.584020
22. Royer, E.M., Perkins, C.E.: 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, pp. 207–218. ACM, New York, NY, USA (1999). DOI http://doi.acm.org/10.1145/313451.313538
23. Wu, C.W., Tay, Y.C.: Amris: A multicast protocol for ad hoc wireless networks. pp. 25–29. MILCOM (1999)
24. Xie, J., Talpade, R.R., Mcauley, A., Liu, M.: Amroute: ad hoc multicast routing protocol. Mob. Netw. Appl. **7**(6), 429–439 (2002). DOI http://dx.doi.org/10.1023/A:1020748431138