Supporting Groupware in Mobile Networks

Nadav Lavi

Supporting Groupware in Mobile Networks

Research Thesis

Submitted in Partial Fulfillment of the Requirements
For the Degree of Master of Science
in Electrical Engineering

Nadav Lavi

Submitted to the Senate of the Technion — Israel Institute of Technology

SIVAN 5765 HAIFA JUNE 2005

The Research Thesis Was Done Under The Supervision of Prof. Israel Cidon and Dr. Idit Keidar in Electrical Engineering Department

Acknowledgment

I would like to thank my family for believing in me and for their endless love and support.

I would especially like to thank my supervisors, Prof. Israel Cidon and Dr. Idit Keidar, for their patience, attention and for not sparing any time and effort for the success of this research.

I thank Prof. Israel Cidon for sharing his academic and industrial knowledge, using his special sense of humor, and teaching me to transform theory into practice.

I also thank Dr. Idit Keidar for driving me towards perfection in a pleasant manner and for teaching me all about research.

I would also like to thank Yoram Orchen for encouraging me to continue my academic experience and study towards an M.Sc. degree, and for providing useful advise whenever I needed them.

Last but not least, to my friends at the Technion, Gal Badishi, Aran Bergman and Isaskhar (Zigi) Walter, a big thank you for the friendship, support and for making my stay at the Technion pleasant and homelike.

The generous financial help of the Anne Abel and Benjamin Pinkel Fellowship is gratefully acknowledged.

The generous financial help of THE TECHNION is gratefully acknowledged

Contents

List of Figures

List of Tables

Abstract

The widespread availability of the Internet has enabled the use of many groupware and collaborative computing applications (chat, ICQ, NetMeeting, Exchange, Lotus Notes, Webex, desktop video conferencing, etc.). With the advance of wireless personal communication, such groupware applications are becoming popular in cellular and mobile networks [35]. For example, major cellular providers (Verizon, Nextel, Orange) offer, or plan to offer soon, group services such as push-to-talk (PTT) [23, 39]. The PTT cellular revenue, which was $84 million in 2003, is expected to reach $10.1 billion by 2008; and the 2.3 million PTT cellular subscribers community of 2003 is expected to grow to 340 million by 2008 [42]. While traditional PTT is limited to voice, the emerging convergence expected in *beyond-3G (B3G)* will merge real-time and non-real time aspects of group communication.

There is strong evidence that future wireless network infrastructure will conform to the TCP/IP architecture and its related supporting mechanisms for real time applications (VoIP, VCoIP), QoS, and mobility. TCP/IP is rapidly being adopted by emerging standards for cellular networks [1, 2, 26], not only at the transport layer, but also at higher level standards such as the session initiation protocol (SIP). This trend enables the convergence of cellular networks with the global Internet [14]. At the same time, low-cost and high-speed wireless access to IP networks is becoming widely available via WiFi access points and WiMAX base stations. Freed from the wire constrains, Internet endpoint devices are becoming smaller, lighter, and easier to operate under mobility conditions. These two parallel trends are leading to gradual convergence between the previously separate worlds of cellular and wireless IP, both at the mobile device level and at the network infrastructure. Given the importance of groupware, the converged wireless network should support cross-network group services for both real-time and data communications.

Consequently, a clear missing link in this evolution is the lack of comprehensive support for group management as well as adequate solutions for real-time applications (QoS, seamless handoff, etc.) in the IP mobility standards. The emerging real-time groupware applications need a solid and integrated framework on which mobile users can be supported.

This thesis presents MaGMA, a novel architecture for group management in mobile networks interconnected via the global Internet. MaGMA provides a comprehensive solution for the mobile world, addressing aspects such as scalable group management, mobility, handoff, and QoS provision. We believe that such a solution will be an inherent part of 3G and B3G cores. We are not aware of any previous comparable solution for mobility support in groupware applications.

MaGMA's architecture consists of a collection of mobile group managers (MGMs), which manage group membership and may also implement a multicast overlay for data delivery. Each mobile node (MN) is served by an MGM proximate to it. Developing a fully distributed group management and mobility solution that is both efficient and scalable is technically challenging. In this paper, we address this challenge.

In this thesis we offer two group management approaches in the context of MaGMA: The first is a *subscription model*, in which the servers (MGMs) support group management only, and MNs can obtain the list of group subscribers in order to transmit data to them without the servers' intervention. The second is the *multicast overlay model*, where the servers implement the group multicast service. The first approach is appropriate for small groups and light servers, whereas the latter is more appropriate for large groups and clients with battery power constrains. We devise protocols for both approaches, and evaluate both solutions using the ns2 network simulator [25]. We validate our simulation results through mathematical analysis. In addition we prove the correctness of the main group-membership protocol under several assumptions. As a proof-of-concept, we also build a SIP-based prototype running groupware applications over WiFi network with iPAQ MNs.

List of Symbols

Abberviations and Acronyms

ARP Address Resolution Protocol

ATM Asynchronous Transfer Mode

B3G Beyond 3^{rd} Generation

CBT Core Based Trees

CN Corresponding Node

CoA Care of Address

DH Domain Hop

DNS Domain Name System

DVMRP Distance Vector Multicast Routing Protocol

FA Foreign Agent

FIFO First In First Out

HA Home Agent

IETF Internet Engineering Task Force

IP Internet Protocol

IPv4 Internet Protocol version 4

IPv6 Internet Protocol version 6

LEC Local Event Counter

MaGMA Mobility and Group Management Architecture

MGCP Media Gateway Control Protocol

MGM Mobile Group Manager

MIP Mobile IP

MN Mobile Node

MOSPF Multicast Open Shortest Path First

MPLS Multiprotocol Label Switching

NGN Next Generation Network

OMA Open Mobile Alliance

PoC Push-to-talk over Cellular

PTT push-to-talk

QoS Quality of Service

RSVP Resource Reservation Setup Protocol

RT Real Time

RTP Real-time Transport Protocol

SIP Session Initiation Protocol

TTL Time To Live

TTS Time To Stabilization

UID Unique ID

VCoIP Video Conference over IP

VoIP Voice over IP

WLAN Wireless Local Area Network

Symbols

Δ Message propagation bound

τ Detection bound of MN's disconnection

$\mathcal{S}$ The set of MGMs in the network

N Number of MNs in the network

D Number of domains in the network

g Number of groups in the network

Chapter 1

Introduction

The widespread availability of the Internet has enabled the use of many groupware and collaborative computing applications. With the advance of wireless personal communication, such groupware applications are becoming popular in cellular and mobile networks.

The converged Internet infrastructure is starting to provide the required support for real-time applications, which require quality of service (QoS) among stationary endpoints. This has led to the emergence of many QoS standards and technologies, e.g., Diffserv, RSVP, and MPLS, as well as real-time protocols such as RTP, H.323, MGCP, and SIP.

At the same time, wireless access to the global Internet is becoming widely supported, and WLAN access points are ubiquitously available. Given the trends predicted in wireless standard forums, it is expected that the next phase in the evolution of converged group services will be their integration with wireless mobile devices. These networks are most likely to adopt the TCP/IP architecture including its related convergence standards.

Currently, there are no adequate solutions to support mobile groupware users. Previous groupware and QoS solutions from the Internet world were not designed with mobility in mind. Mobile IP (MIP), the proposed IP mobility solution from the IETF, incorporates large overhead and lacks QoS support. Other groupware solutions focus on ad-hoc networks or do not address real-time (RT) communication and QoS. Current cellular solutions are insufficient for mobile RT clients.

This thesis presents MaGMA, a comprehensive solution for the mobile world, addressing aspects such as scalable group management, mobility, handoff, and QoS provision. The main goals of MaGMA are mapping group names to their current subscribers, supporting mobility with seamless handoffs, QoS support for RT applications, transport efficiency including the avoidance of triangle routing and a scalable control plane. MaGMA's architecture consists of a collection of servers, named mobile group managers (MGMs), which manage group membership and may also implement a multicast overlay for data delivery. Each mobile node (MN) is served by an MGM proximate to it. MaGMA offers two group management approaches: the subscription model approach, supporting direct end-to-end communication between the sender and the receivers, and the multicast overlay model approach, supporting large groups and enhanced QoS scheme.

In the subscription model, the MGMs manage the group membership and do not participate in data delivery to the MNs. When an MN sends a messgae to the group it first retreives the group information from the MGM incharge of its domain, and then it sends the message directly to each and every MN that is a member of the group. This model that incorporates a peer-to-peer (P2P) approach and therefrore demands the use of MNs level membership information, meaning every MGM that has MNs that joined the group, holds the information of all the MNs in the network that participate in the group. The subscription model provides a P2P way of communication among group members, the most popular communication approach in the Internet today. It is well suited for small groups, where the sender needs to communicate with a small number of receivers, and for light servers, as MGMs only manage group membership. The model is not appropriate for large groups or for battery constrained nodes. In addition, in large groups the subscription model can load the network, as the number of MNs located in a domain and participating in a group grow the number of incoming data streams grow respectively.

The multicast overlay model enhances the service by delegating data delivery to the MGMs. In this model, when an MN sends a message to the group, it sends a single copy of the message to the MGM in charge of its domain. The MGM then forwards the message to the rest of the MGMs participating in the group, which in turn forward the message to MNs in their domains that participate in the group. In this model, MGMs represent their local MNs and therefore hold two levels of group information: one of the local MNs participating in the group, and the second of other MGMs participating in the group. The multicast overlay model provides a scalable and flexible solution for both group membership and data delivery.

Although the subscription and the multicast overlay models use different approaches of group membership and data delivery, they can coexist in the same network: different groups can use different solutions, depending on the group's characteristics. In this thesis, we only provide a general overview of the subscription model, and then give a detailed and formal description of the multicast overlay model, including pseudo-code and a formal correctness proof of the group membership protocol.

This thesis is organized as follows. In Chapter 2 we describe previous work on both mobility and group membership, and emphasize the advantages of MaGMA compared to the mentioned solutions. Chapter 3 describes the network model and MaGMA's architecture. Chapter 4 provides a general description of the subscription model, and Chapter 5 gives a detailed description of the multicast overlay model. In Chapter 6, we evaluate MaGMA's control plane using simulations conducted in ns2 as well as mathematical analysis. In Chapter 7, we evaluate the performance of the data plane using ns2 simulations and compare it to Mobile IP. In Chapter 8, we describe the implementation of a simple prototype of MaGMA. Chapter 9 proposes several possible future enhancements of MaGMA, and Chapter 10 concludes. Appendix A provides analysis of the subscription model. Appendix B compares the data planes of the subscription and multicast overlay models, and Appendix C provides the correctness proof of the multicast overlay model.

Chapter 2

Literature Survey

We now overview previous work that addresses the various issues addressed by MaGMA: Section 2.1 discusses mobility and groupware solutions for cellular networks, Section 2.2 discusses leading mobility solutions for the IP world. Section 2.3 discusses IP multicast solutions and Section 2.4 discusses multicast support in MIP. Section 2.5 discusses related group communication systems used in IP networks. We observe that these previous solutions and combinations thereof do not provide comprehensive and scalable support for all the needs of real-time mobile groupware applications, namely mobility, group management, and QoS.

2.1 Cellular Mobility and Groupware Solutions

3GPP and 3GPP2 [1, 2] networks are based on the mobility concept of SIP Mobility (described in Section 2.2). When a cellular user (also named MN) is located in its cellular provider's network, incoming calls are routed through the provider's core to the MN. When the MN moves (roams) to another cellular provider, it receives a temporary cellular-number that represents the MN while residing in this foreign cellular network. The foreign network informs the user's home network regarding the new temporary number of the user. When a corresponding-node (CN) initiates a call towards the roaming MN, it receives from the MN's home network the temporary number of the MN at the foreign networks. Then the CN redirect the call towards the current location of the roaming MN. Currently 3GPP and 3GPP2 networks do not support smooth handoff between cellular provides, i.e. the MN disconnects itself from one network and only then connects to another network. Furthermore, this scheme does not support simultaneous movements and may suffer from session initiation delay, depending on the proximity of the MN to its home.

Current industrial solutions for PTT in cellular networks [24, 18, 37] implement the OMA-PoC [26] standard, which uses a centralized server. The server is in charge of both group management and data replication. This solution, although used by various cellular operators, suffers from lack of scalability and excessive end-to-end data delay since it routes all data through the centralized server. We note that this scheme is similar to Mobile IP (described in Section 2.2) and its triangle-routing concept.

An alternative approach for supporting instant messaging and chat in the wireless world is proposed by Jabber [15]. Jabber's solution uses a distributed architecture of servers and e-mail-like addressing. Due to the use of such addressing, data is always sent through the home servers of both the sender and the recipient, which is not the optimal route at times of mobility, and can degrade the performance of real-time applications.

2.2 Mobility Solutions for IP

Mobile IP (MIP) [27] is the current mobility standard for IPv4. MIP was developed to eliminate the need for IP stack modifications in the emerging mobile networks. When an MN moves from its home domain to another, it changes its point of attachment and therefore its IP address. If during this movement the MN is communicating with another node, named CN, the communication session will be disconnected as the CN continues to send messages to the IP address of the MN at its home domain. MIP solves this problem using simple tunneling. Every MN is associated with a home domain and a server in that domain that acts as its *home agent* (HA). While at its home domain, the MN receives packets using the usual IP mechanisms, like a stationary node. When the MN moves to a foreign domain and changes its IP address, it notifies its HA of its new IP address through a *foreign agent* (FA) located in the domain. The FA functions as the *care-of address* of visiting MNs. The HA then forwards to the MN packets destined to the MN's home IP address through a tunnel it creates to the FA. This forwarding scheme, called *triangle routing*, often leads to inefficient routes, and creates a strong dependence of the MN on its home. In addition, the tunneling scheme incorporates large IP header overhead due to the use of IP-in-IP encapsulation. The MN can send the CN messages through triangle route via the FA-HA tunnel, as described in [19], which incurs an additional delay. Alternatively, it can send the messages directly to the CN, using address spoofing. The main problem with this solution is that it does not comply with current security recommendations, which recommend that every domain should discard spoofed messages in its root router and should not route these messages to the Internet. Although MIP does not support group communication, as MaGMA does, we compare MIP to MaGMA as current industrial solutions use MIP's triangle-routing concept to route data to MNs. We note that MaGMA does not use triangle routing as it eliminates the home concept.

Perkins and Johnson [29] suggest route optimizations to Mobile IP, which avoid triangle routing. In this approach, the HA sends *binding* information to the corresponding node, thus enabling direct communication between the two. It is unclear whether Mobile IP with route-optimizations can support simultaneous movements of both endpoints of a communication. Moreover, this solution requires a modification of the host IP stack, and a home agent in each MN's home domain, and may therefore be difficult to deploy. In addition, establishing new connections to an MN always involves its home domain, even if the MN is distant from it for an extensive period. In contrast, MaGMA provides a flexible architecture in which the communication infrastructure is deployed at the core network, and communication with an MN is completely independent of its home domain.

Balakrishnan and Snoeren [32] propose a DNS-based solution to IP mobility. Similar to Mobile IP, every node has a home domain (or DNS zone). When an MN moves and changes its IP address, it registers a secure DNS update at its domain DNS server. In order to avoid the use of stale binding information, DNS caching is minimized (by setting TTL=0) and direct binding is used. The major drawbacks of this approach are that both endpoints cannot move simultaneously, that DNS standards do not support the proposed user self-configuration, and that operating systems and DNS servers often do not comply with DNS TTL caching directions. Finally, eliminating DNS caching is bound to overload the DNS system.

Wedlund and Schulzrinne [41] use the Session Initiation Protocol (SIP) [31] to support RT communication between mobile users. Similar to Mobile IP with route optimizations, nodes update their home (SIP proxy) regarding their new locations. When a node invites another node (using an INVITE message) to initiate a session, it receives a redirection message from the SIP proxy. This way, it can contact the moving node directly. If a node moves during an ongoing session, then it sends a new INVITE message with the session identifier to the other node. This solution, like Mobile IP with route optimizations, establishes a connection through the home domain. Thus, the session initiation delay depends on the proximity of the MN to its home. In addition, [41] does not address scenarios where both hosts are mobile and move during a session.

IPv6 [10] is the new IP protocol stack that aims to remedy all illnesses of the IP world, including IP mobility. The main advantages of IPv6 are the 128 bits address space, which allows for a large number of mobile nodes, and the flexible way to implement new protocols, such as Mobile IPv6, using extensions to the option header, which is part of the IPv6 header.

Mobility support in IPv6 [16, 28] is based on the MIP concept and enhanced by the new features of IPv6. New concepts implemented in IPv6 enable better support in MNs, for example the acquisition of a new CoA is done using DHCPv6 [11] or using the enhanced IPv6 Host Address Auto-Configuration [36], which combines DHCP, ARP, and neighbor discovery. Using the possibility to expand the IP header, MIPv6 eliminates the need for the foreign agent. This is done using the mobility header extension to the option header and the routing header. Using the mobility header, the MN sends binding information to its HA and to the CN, and using the routing header, more optimized routing is achieved. In addition, mobility in IPv6 is supported using reverse tunneling and direct binding. This way, mobility is transparent for both mobility aware CNs and for regular CNs with no mobility support. Although IPv6 and MIPv6 can provide better support for communication with MNs in the future mobile next generation network (NGN), it is not widely deployed. Moreover, it is clear that during the next years, during the transition from IPv4 to IPv6, most of the IP network will be based on IPv4 with small islands of IPv6. And even in the future, when IPv6 will be well deployed, IPv4 networks will still be in used. Therefore there is a need for a comprehensive solution supporting both versions of IP bridging between the two networks, as MaGMA does.

2.3 IP Multicast

In [13] several multicast schemes supporting mobile users are described. Most of the multicast protocols, such as DVMRP [40], CBT [5, 4] and MOSPF [20], were developed to address multicast in a stationary environment. Newer multicast schemes, such as Mobicast [34], improve the multicast scheme to support mobile users by enhancing micro-mobility.

Mysore and Bharghavan [21], suggest the use of the IP multicast infrastructure in order to support host mobility. In this solution, each MN is assigned a special multicast address, and when the MN moves, it causes the multicast tree to change.

While these solutions can potentially provide good performance, unfortunately, IP multicast is not widely deployed. Therefore, these schemes, supporting group communication or mobility, cannot provide seamless mobility in today's Internet. Finally, groupware applications may combine both multicast and unicast communication and therefore combining the groupware management and transport into one multicast mechanism may be very restrictive.

2.4 Multicast Support Using MIP

Two solutions to support multicast using the MIP standard are presented in [27]. One solution is based on the local registration of an MN in foreign domains. This way, the foreign agent functions as a multicast router and forwards to the registered MN the multicast group packets. In the second solution, the MN sends a registration message to its HA, which in turn creates a tunnel towards the FA, and forwards the multicast group packets to the MN through the FA using encapsulation. Both solutions suffer from scalability problems. The first solution may suffer from packet loss due to the long set-up time associated with multicast registration. The second solution suffers from large overhead due to the encapsulation and from packet duplication when several HAs create tunnels associated with the same multicast group toward the same FA. There are several additional problems and issues as described in [9]: one problem is the duplication, where duplicated multicast packets enter the domain when both a local node and a roaming node join the same group. Another problem is scoping, where the existence of several small-scope groups that have the same address in the network cause nodes to receive multicast packets that originally where destined to other recipients. In [9], the authors propose an enhanced architecture, based on the tunnel solution. Although this architecture solves several of the problems discussed above, it incurs additional control overhead, and still suffers from the encapsulation overhead and from the long delays of triangle routing. Thus, this architecture is inadequate for RT applications.

2.5 IP Group Communication Solutions

CONGRESS [3] is a scalable group management protocol, which was designed for ATM environments. Like MaGMA, CONGRESS uses a server overlay. However, CONGRESS is restricted to hierarchical overlay while MaGMA is more flexible and not restricted to a certain overlay construction. Like MaGMA's subscription model, CONGRESS only address membership management, and does not support QoS multicast. Moreover, CONGRESS was not designed with mobility in mind and does not incorporate a handoff solution. In conclusion, CONGRESS does not provide a comprehensive solution for group management, multicast, and mobility as MaGMA's multicast overlay model does.

Most IP-based group communication systems have not addressed mobility. The only exceptions that we are aware of are [30, 6]. Prakash and Baldoni [30] propose protocols for group communication based on an a priori knowledge of the group members combined with a proximity protocol to discard group members located more than D hops from the sender. In addition, they focus on virtual cellular networks, where base stations can move, and ad-hoc networks where there are no base stations. In contrast, we consider a more prevalent network topology, where there are stationary base stations. Focusing on a less difficult problem allows us to develop more efficient solutions. Moreover, our solution proposes a dynamic group membership where MNs can join and leave the group. Bartoli [6] proposes a centralized totally-ordered multicast protocol for a dynamic membership in wireless networks.

Chapter 3

Network Model and Architecture

3.1 Network model

We model the network as a collection of autonomous domains. Every MN has a unique ID (UID), which identifies the MN at all locations. Upon moving to a new domain, the MN obtains a new local IP address (e.g., using DHCP). We assume that a micro-mobility mechanism is in place in each domain (whether cellular or WiFi), and that an adequate IP routing protocol exists in each domain.

We assume that message propagation time is bounded by some Δ time units and that all the network channels support reliable FIFO communication. We further assume that the network supports smooth handoff, and that the handoff delay is negligible, that is, an MN detects that it enters a new domain immediately. When an MN moves to a new domain, it does not move again to yet another domain for at least 10Δ time units. This movement bound is more than reasonable, for example if $\Delta = 100msec$ (a reasonable end-to-end delay in a network that support RT communication [12]), we expect the MN to stay in a domain for at least a second. In addition, we assume that MGMs do not crash and that MNs can crash, and such crashes are detectable: when an MN crashes, its local MGM detects the crash by $2\Delta < \tau < 3\Delta$ time units.

3.2 The Architecture

MaGMA consists of MGMs distributed throughout the network. Ideally, an MGM is located in each domain. For the sake of simplicity, we assume that the MGMs are static and well-known. The MGMs form an overlay network among them. We propose two approaches to support groupware: in the subscription model, the MGMs are only in charge of group management, whereas in the multicast overlay model, they are in charge of both group management and data delivery over the overlay network. The overlay construction can employ known techniques for building efficient QoS-aware overlays, e.g., [33, 8], and is beyond the scope of this paper. In the multicast overlay model, MGMs distribute traffic within their domains to the appropriate MNs. To this end, they can use either unicast or multicast– we recommend the use of multicast where available as it is more efficient.

MaGMA can be used in several ways to allow interoperability among WiFi and cellular users. For example, the cellular provider can implement a single MGM at the cellular core managing the entire cellular network, or a distributed MGM system where each MGM manages a domain of several base stations. Other MGMs can be located at WiFi/WiMAX domains. An example of such an integrated architecture is depicted in Figure 3.1.

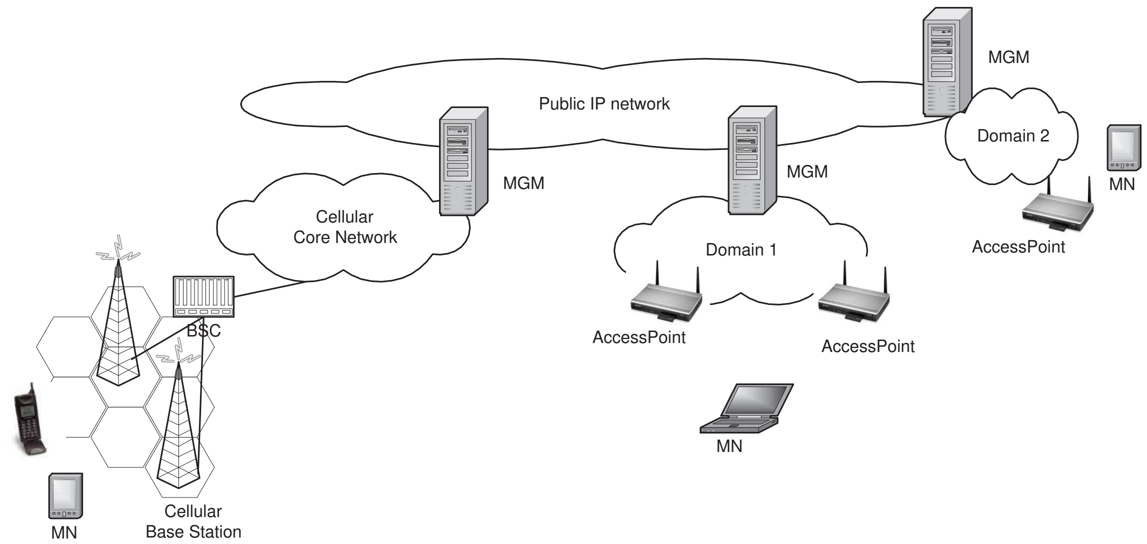


Figure 3.1: An example integrated network architecture.

MaGMA calls for the use of distributed servers for the following reasons:

- to offer scalability in the number of groups and the number of group members;
- to efficiently support groups with geographically dispersed members;
- to facilitate QoS reservation among domains;
- to reduce traffic overhead; and
- to enable vertical handoff between networks.

Each MN is served by the MGM closest to its domain.

The MGMs provide to the MNs group services such as *joining* and *leaving* a group, and they enable MNs' movement between domains. In the subscription model, MNs can retrieve group views and receive continuous reports of membership changes from the MGMs. In the multicast overlay model, MGMs provide multicast and data delivery services. In addition, MGMs may provide advanced application support services, which are left for future research.

We assume that MNs are likely to remain in groups for considerable periods. Therefore, we assume that move operations dominate the control traffic.

Chapter 4

MaGMA Subscription Model

In the subscription model, MGMs manage MN-level group membership. That is, they keep track of which MNs are in the group at any given time. We present two solutions for the subscription model. We begin, in Section 4.1, by presenting a simple solution in which all MGMs keep track of all the groups, regardless of whether they have group members. Then, in Section 4.2, we present an optimized solution where only MGMs that have group members are involved in managing the group. Note that both solutions may coexist in the same network for different group types.

4.1 MGMFlood

In our first scheme, MGMFlood, each MGM forwards (floods) to all other MGMs all control messages (*join/leave/move*) received from MNs in its domain. When an MN crashes, its local MGM detects the crash and sends an appropriate *leave* message to all other MGMs.

MGMFlood is simple and allows for seamless handoff due to its prompt reaction to mobility updates. However, it entails high control message overhead, as all MGMs keep views of all groups, including groups not residing in their domains. This solution may be appropriate for managing very large and geographically diverged groups, but it does not scale well, especially in the case of many small groups and localized memberships.

4.2 MGMLeader

Our second solution reduces the overhead by propagating updates only to those MGMs that have group members in their domains. When an MGM receives an MN's message regarding a group that is represented in its domain, it extracts the MGMs that have members in the group from its local view, and forwards the message only to those MGMs.

If an MGM receives a control message (*join* or *move*) for a group that does not yet exist in its domain, then it needs to discover the group's up-to-date view, and to forward the event to the appropriate MGMs. The challenge is preserving a coherent view at all MGMs in the presence of concurrent operations without inducing excessive overhead.

In order to minimize the control overhead and ensure view consistency, only one of the participating MGMs sends the view to the new MGM. To this end, one MGM is designated as the *coordinator* of the group. Every active group has a coordinator, and a single MGM can be the coordinator of multiple groups. If the coordinator leaves the group (because all the MNs in its domain leave) then a new coordinator is elected, as explained Section 4.2.2 below.

We now proceed to describe our protocol. We present a general description of the protocol without pseudo-code only in order to provide intuition for the issues and solution techniques. In the next chapter, we formally describe an extension of this protocol for the multicast overlay model. In Section 4.2.1, we explain how the coordinator manages the group while there are no coordinator changes. In Section 4.2.2, we discuss how a new coordinator is elected when there is none, and in Section 4.2.3, we describe the coordinator transition process. In Section 6.2 we evaluate the control overhead using both mathematical analysis and simulations.

4.2.1 Normal Operation

When a new MGM joins a group due to a *move* event, it extracts the moving MN's former MGM from the *move* message, and sends the event message to that MGM. The former MGM, in turn, forwards the message to the coordinator. When the coordinator receives a *move* message originating from an MGM that is not already in the group, it sends the group's view to the new MGM and forwards the message to all the group's MGMs. This message flow is illustrated in Figure 4.1. This communication between the two MGMs also facilitates establishing a tunnel from the former MGM to the new one, so that the former MGM can forward data packets destined to the moving MN via its new MGM, to guarantee smooth handoff; such tunneling is suggested in [29].

When a new MGM joins a group due to a *join* message, it has no knowledge of which MGMs are currently in the group. Therefore, the joining MGMs broadcast a join message to all the MGMs in the system. As before, when the coordinator receives this message from the new MGM, it sends the group's view to the MGM. In this solution, the cost of a new MGM joining the group is high. An improved join mechanism reducing this cost is described in Section 4.2.4.

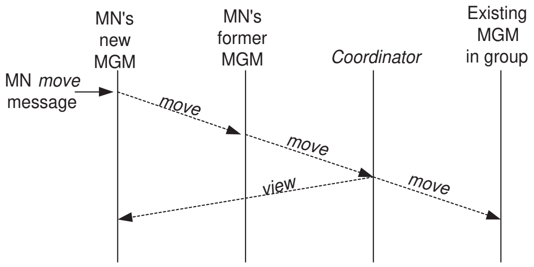


Figure 4.1: *Move* message flow.

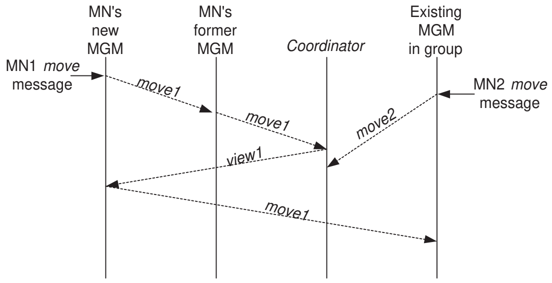


Figure 4.2: Potential view inconsistency in over-simplified leader-based solution.

Different MGMs may receive certain event messages in different orders. Thus, there is a need to ensure view consistency when MGMs dynamically join and leave the group. Figure 4.2 illustrates a problematic scenario that can occur if concurrent joins are handled carelessly. In this example, while a new MGM retrieves the group's view from the coordinator, an existing MGM sends another event to the group's MGMs. The existing MGM is unaware of the new MGM and thus does not forward the message to it. This causes the new MGM to have an inconsistent view of the group.

In order to address this difficulty, each MGM maintains an increasing *Local Event Counter* (LEC) for every group. Whenever an MGM receives a *join*, *leave*, or *move* message from a local MN, it increments the appropriate LEC. The group's LEC is included in every message pertaining to this group sent by the MGM. When an MGM joins a group, it initiates the group's LEC to 1. In addition, the MGM keeps, for every active group, a *LECvector*, holding the highest known LEC for each MGM in this group.

In every message sent from one MGM to another, both the sender's LEC and the receiver's latest known LEC (from the *LECvector*) are included. When an MGM receives a packet, it checks the LECs. If its local LEC is higher than the one known to the sender it sends back its local view and LEC. If it discovers that the sending MGM's LEC is higher than the one it knows, it retrieves the local view of the sending MGM. When the coordinator forwards *move* messages of new MGMs, it includes the LECs corresponding to the view it is sending to the new MGM. In case some events are not reflected in this view, the receiving MGMs forward their local views to the new MGM.

4.2.2 Election Procedure

When an MGM first joins a group, it has no information regarding the group members and its coordinator. Therefore, it broadcasts to *all* MGMs a *join* message that includes the group name. In case a coordinator is active, it replies with a *view* message conveying to the joining MGM the necessary group information (the members and the coordinator's UIDs). If a coordinator does not exist, i.e., there are no MGMs participating in this group, then this broadcast initiates a coordinator election procedure.

The election procedure can be initiated by several MGMs that almost simultaneously attempt to join the group. We must ensure that all MGMs participating in the election procedure have the same view of which other MGMs are participating. In order to keep track of the participants, an MGM that sends a *join* message buffers all the *join* messages it receives during a period of 3Δ time units after its own broadcast. This guarantees that MGMs wait long enough in order to get the coordinator's response during a coordinator transition, as explained in Section 4.2.3 below. During this period, new messages received from MNs are also buffered; the MGM processes these messages only after the coordinator election ends. MGMs that do not send a *join* message before receiving another MGM's join cannot join the group for a period of 3Δ time units. During this time, all messages received from MNs are buffered; these messages are processed only after this guard time ends. In order to prevent inconsistencies in the election of a coordinator, the guard timer must be restarted upon reception of every new *join* message.

When the guard time interval ends, MGMs that participate in the election procedure elect an MGM from among the MGMs whose *join* messages have been received to be the group's coordinator. The election can be based on UIDs or any other parameter included in the *join* messages, e.g., which MGM serves the highest number of MNs, or administrative priority. When the guard time expires at an MGM that does not participate in the election, this MGM can join the group (if necessary) by broadcasting its queued *join* message.

We illustrate the need for a correct measurement of the guard-time in Figure 4.3. In this scenario, MGM3 and MGM2 first initiate the election procedure. MGM1 receives the *join* message of MGM3 and starts the 3Δ guard time. In Figure 4.3(a) we show what happens if MGM1 does not restart the guard time when it receives MGM2's *join* message. When the guard time ends, MGM1 joins the group (due to an MN's *join* message). In this case, MGM2 receives the message before t_2, which is the end of its 3Δ guard time, whereas MGM1 does not receive the join before t_1, which is the end of its guard time. Thus, they elect two different coordinators. In Figure 4.3(b), MGM1 correctly restarts the guard time when it receives MGM2's *join* message. Thus, MGM2 receives MGM1's *join* message only after the end of its guard time, and it does not influence its election of the coordinator.

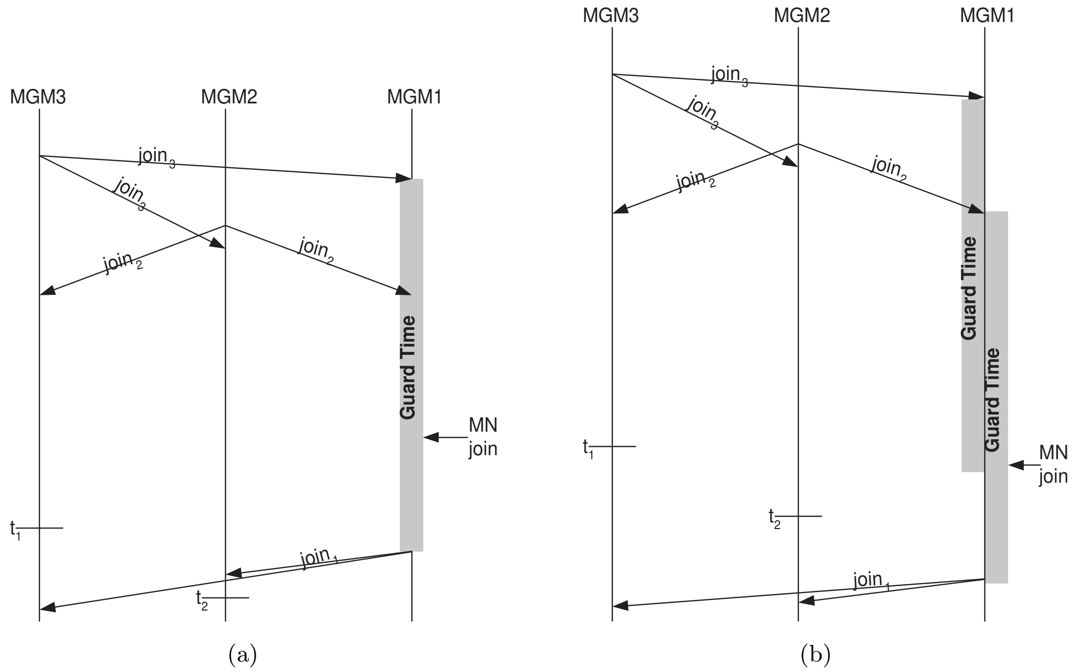


Figure 4.3: Illustrating the need for restarting the guard: (a) an incorrect election procedure without restarting; and (b) a correct election procedure.

4.2.3 Coordinator transition

When the last MN in the coordinator's domain leaves the group or moves to another domain, the coordinator appoints a new MGM as the new coordinator of the group and informs the group's MGMs of the new coordinator in the forwarded *move* or *leave* message. Subsequently, the leaving coordinator creates a tunnel to the chosen coordinator, and forwards control messages that it still receives on this tunnel. In order to avoid appointing an MGM that has already left, an MGM can not leave the group until it receives the coordinator's permission. If the coordinator notices, after receiving a *leave* or *move* message, that an MGM has no members in the group, it sends a *permission-to-leave* message to that MGM. The only scenario where the coordinator does not permit the MGM to leave when its group empties is if the coordinator has already appointed the MGM to be the new coordinator. In this case, the chosen MGM needs to find a new MGM to replace it as the group's coordinator. When the group is empty, i.e., there are no MNs participating in the group, the coordinator can leave the group without informing other MGMs.

We now explain the selection of the 3Δ time units guard time in the election procedure. When a new joining MGM sends a *join* message for an active group, the group's coordinator may be in transition. In this case, the coordinator does not reply with the group's view immediately. When the transition ends, (i.e., the new coordinator receives the *transition* message), the new coordinator starts handling messages received both directly and via the tunnel from the previous coordinator. Therefore, the *join* sent by new joining MGM is received by the new coordinator at most 2Δ time units after being sent, and thus, the joining MGM receives the group's view from the coordinator within 3Δ time units.

According to the guard-time and the transition time, we calculate the required bound on MN movements. As explained above, when an MN moves into a new domain, the handoff mechanism exploits the fact that the previous MGM holds the coordinator information. To ensure that the handoff mechanism will work properly, we use a bound on MN movements, requiring that an MN will not move to another domain before its current MGM completes its joining procedure and receives the group's view and coordinator information. This way, *handoff* messages can be forwarded to the coordinator within 2Δ time units. We now examine the worst case scenario of the joining procedure in terms of the time it takes the MGM to set the view. The scenario is illustrates in Figure 4.4.

Let T_{guard} be the guard-time as described in Section 4.2.2. Let $T_{respond}$ be the time that elapses since a coordinator (active or previous) receives a *join* message from an MGM and until the MGM receives a view back from the coordinator. If no transition is taking place, this takes Δ time units. In case of transition, it takes Δ to tunnel the message to the new coordinator, and Δ more from the new-coordinator to the MGM. Thus, in this protocol, $T_{repond} = 2\Delta$. As illustrated in Figure 4.4, an election begins when MGM1 sends a *join* message at time t. MGM3 receives the message almost immediately, and due to the fact that it does not participate in the group it enters into a guard interval for T_{guard} time units. Right after that, MGM3 receives a *join* message from an MN located in its domain, but MGM3 can send a *join* message only after the guard-time ends. MGM2 joins the group as well, and MGM3 receives its *join* message at $t + 2\Delta$. Therefore, MGM3 resets its guard timer at $t + 2\Delta$ and broadcasts its *join* message only at $t + 2\Delta + T_{guard}$. MGM3 cannot receive additional *join* messages after $t + 2\Delta$, as all MGMs receive MGM1's *join* message by $t + \Delta$, and therefore MGMs that did not send their *join* messages before that time have to wait at least T_{guard} time units before they are allowed to broadcast their *join* message. Therefore, $t + 2\Delta + T_{guard}$ is the latest time in which an MGM can still have a guard from this election. The group's coordinator receives the message at $t + 3\Delta + T_{guard}$, and sends back the view. MGM3 receives the *view* message by $T_{respond}$. Therefore, only after $3\Delta + T_{guard} + T_{respond}$ time units, after receiving the MN's *join* message, MGM3 receives the group's view and coordinator information. In the subscription model, $T_{guard} = 3\Delta$, and $T_{respond} = 2\Delta$. Thus, after 8Δ time units MGM3 receives the group's view. Using the

10Δ time units movement bound, as described in Section 3.1, we allow a successful handoff procedure in case the MN moves into another domain.

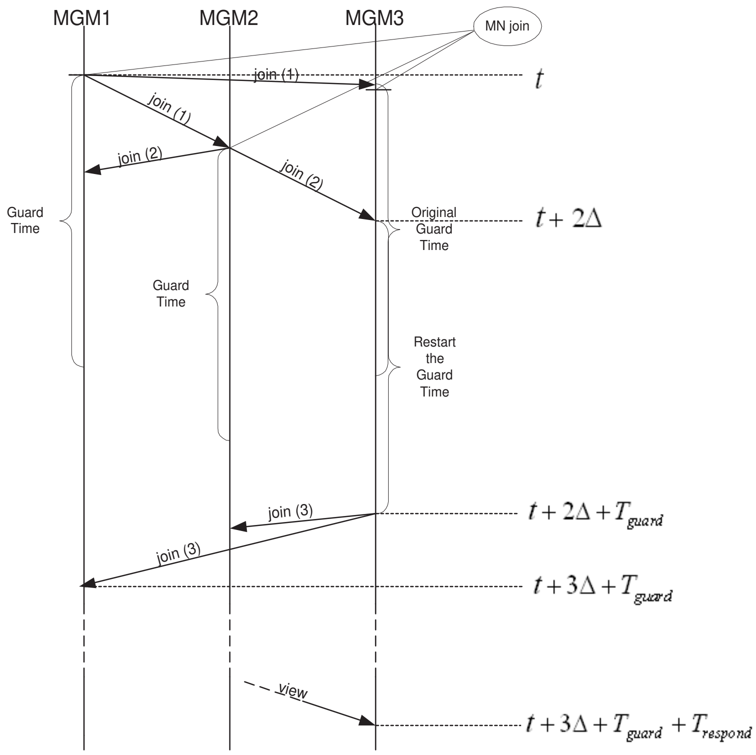


Figure 4.4: Illustration of the maximal response time of the coordinator to an MGM join.

4.2.4 Improved Join Mechanism

We now describe an approach, based on hash functions, to reduce the number of messages sent per join. In this approach, there is a well known hash function mapping a group ID to an MGM UID, using the hash function every group is mapped to a single MGM that represents it. The role of this MGM is to hold the UID of the current coordinator of the group. This way, new joining MGMs do not need to broadcast their *join* message to the entire network. Instead, they can query the appropriate MGM, according to the group ID and the hash function.

When an MGM that represents a group receives a *join* message, it checks if the group is already active. If it is active, i.e., there is a functioning coordinator, it forwards the *join* message to the coordinator that handles the joining procedure as described above. If the group is not active, the representing MGM informs the new joining MGM that it is the group's coordinator, and saves its identity. It is clear that this method out-performs the election procedure described in Section 4.2.2, as there is no need to use broadcast and to wait for 3Δ time units before electing a coordinator.

When a coordinator leaves the group, it has to inform the new coordinator, the rest of the MGMs participating in the group, and the representing MGM regarding the new elected coordinator. When a coordinator leaves an empty group, i.e., there are no other MGMs participating in the group, it requests the representing MGM a permission to leave the group.

Chapter 5

MaGMA Multicast Overlay Solution

In this chapter, we discuss an alternative MaGMA solution, in which the MGMs provide the multicast service. In this approach, MNs do not obtain the list of subscribers in the group in order to send messages to them, but instead hand multicast messages over to their MGMs. Therefore, there is no need for each MGM to track the full list of subscribers in each group and their addresses. Rather, each MGM needs to know the identities and addresses of its *local* subscribers, (i.e., MNs in its domain), as well as which other MGMs have members in the group. MNs in other domains are represented by their domains' MGMs. In other words, the control plane of the multicast overlay solution maintains an MGM-level membership and a local MN-level membership, but not a global MN-level membership as the subscription model solutions do.

We present a protocol for this model based on the MGMLeader approach described above. We now overview the group management protocol and present a detailed description and pseudo-code, the protocol's correctness proof is presented in Appendix C.

An MGM participates in the protocol if it manages at least one MN that belongs to this group. In addition, the MGMs participate in the distribution of data. When an MN needs to send data to a group, it forwards the data packets to its MGM indicating the destined group. The MGM forwards the packets to the other MGMs that participate in the group via the overlay. When an MGM receives data packets from another MGM, it forwards the packets to its local MNs that participate in the group.

An MGM that is not a member of a group joins the group either when an MN in its domain sends a *join* message, or when an MN that already participates in the group moves into its domain and sends a *move* message to inform the MGM of its arrival.

An MN sends a *leave* message to the MGM when it wishes to leave the group. An MGM receiving this message removes the MN from its local view. If the local view becomes empty, i.e., no local MNs are group members, the MGM initiates a procedure for leaving the group.

5.1 Data Structures and Message Structures

Every MGM and MN holds a unique ID (UID), MGMs also manage an increasing *Local Event Counter* (LEC). The LEC is initialized to 1 and incremented upon local join, leave, and move events, and during transition. In addition, MGMs hold the data structures described in Table 5.1.

Data Structures			
Data Structure	**Description**	**Type**	**Init**
groupView	MGMs participating in the group	set of $\langle MGM, LEC\rangle$	$\emptyset$
localView	MNs participating in the group	set of MNs	$\emptyset$
joiners-list	MGMs participating in the election procedure	set of $\langle MGM, LEC\rangle$	$\emptyset$
coord	UID of the group's coordinator	MGM	$\perp$
next-coord	UID of the next coordinator in case of transition	MGM	$\perp$
active-tunnels	MGMs that joined up to Δ time units ago	set of $MGMs$	$\emptyset$
msgBuffer	received messages	set of msgs	$\emptyset$
abstinence	Indicating on election participation	{true, false}	false
leaveFlag	Indicating if an MGM can leave	{true, false}	true
$lec(groupView, i)$	returns the LEC of an MGM from *groupView*	func:(set of ⟨MGM,LEC⟩,MGM)→ LEC	
Externally Provided Function			
coordSelect	determenistic function for coordinator selection	func:set of MGMs→ MGM	

Table 5.1: Data structures and functions used by MGMs.

The formats of the messages structure sent by the MGMs are described in Figure 5.1.

5.2 *LEC* Verification

When an MGM receives a message from another MGM, it verifies that the *LEC* included in the received message is higher than the one stored in its view. If the *LEC* is lower, the

join: $\langle MGM_i, join, LEC_i \rangle$
leave: $\langle MGM_i, leave, LEC_i \rangle$
handoff: $\langle MGM_i, handoff, LEC_i, UID_{MN} \rangle$
view: $\langle MGM_i, view, groupView \rangle$
transition-request: $\langle MGM_i, transition\text{-}req, groupView, LEC_i \rangle$
transition-granted: $\langle MGM_i, transition\text{-}granted, LEC_i \rangle$
transition-denied: $\langle MGM_i, transition\text{-}denied, LEC_i \rangle$
new-coord: $\langle MGM_i, new\text{-}coord, LEC_i \rangle$

Figure 5.1: Message types (MGM$_i$ is the sender).

receiver discards the message. Otherwise, it updates the *LEC* in the view and processes the message according to its type as described below. The *LEC* verification is shown in Figure 5.2 Lines 2–5.

1: **upon** receiving msg $\langle ..., LEC_k \rangle$ from MGM_k **do**
2:   **if** $LEC_k \leq lec(groupView, k)$ **then** {*LEC verification*}
3:     discard *msg*
4:   **else**
5:     $lec(groupView, k) \leftarrow LEC_k$
6:     **if** *next-coord*$\neq \perp$ and *coord*$= i$ **then** {*tunneling*}
7:       forward *msg* to *next-coord*
8:       add *msg* to *msgBuffer*
9:     **else if** *active-tunnels*$\neq \emptyset$ and *coord*$= i$ **then**
10:       forward *msg* to *active-tunnels*
11:     **else if** *localView*$\neq \emptyset$ **then**
12:       process msg according to type

Figure 5.2: Message pre-processing.

5.3 Joining a Group

An MGM joins the group by broadcasting a *join* message to all MGMs in the network. At the same time, the MGM initiates a guard-timer Figure 5.3 line 5. This timer period is used for the bootstrapping procedure and coordinator election in case no coordinator exists at the time of join. The coordinator election procedure is similar to the one presented in Section 4.2.2. The only differences are that the guard time is 4Δ time units instead of 3Δ as above. The difference in the guard time stems from the different transition procedure employed in this protocol, as described in Section 5.4 below. As joining is not synchronized among the MGMs, several MGMs can join simultaneously. During the timer period, the joining MGM stores in *joiners-list* UIDs of other MGMs that send *join* messages. If the group is active, i.e., a coordinator exists, it sends back the group's view, Figure 5.4 line 5, before the timer expires and the MGM stops the bootstrapping procedure (Figure 5.5). If the group is inactive, after the timer expires, the joining MGM can elect a coordinator according to the UIDs stored in *joiners-list* and the deterministic function *coordSelect* and assign *groupView* to hold the identities of the MGMs stored in *joiners-list*, i.e., the MGMs that participated in the election (Figure 5.3 lines 7–8). After setting the coordinator and the group's view, due to the end of the election procedure or due to the reception of *view* message from the coordinator, the joining MGM flushes its *joiners-list*. To avoid inconsistency, an MGM not participating in the group that receives a *join* from another MGM will not join the group for 4Δ time units (by turning on its *abstinence* flag for 4Δ time units).

Due to the fact that MaGMA multicast overlay is based on MGMLeader it is possible to improve the joining procedure using the method described in Section 4.2.4. In this method a joining MGM sends its *join* message to a representing MGM, selected according to a well-known hash function. This MGM holds the UID of the group's coordinator and forwards *join* message to that coordinator, if it exists, and elects a new coordinator if the group is not active. This method decreases the number of messages sent during a join procedure, as it eliminates the need for broadcast, and prevent the need to wait for 4Δ time units during an election, as the representing MGM appoints the group's coordinator.

```
1:  upon joining group do
2:    wait until !abstinence
3:    broadcast ⟨join, LEC⟩
4:    add ⟨MGM_i, LEC_i⟩ to joiners-list
5:    wait 4Δ time units or until processing view message
6:    if coord=⊥ then
7:      coord = coordSelect(joiners-list)
8:      groupView ← joiners-list
9:      if coord=i then
10:       set leaveFlag←false for Δ time units
11:   joiners-list = ∅
```

Figure 5.3: MGM join code.

```
processing ⟨MGM_k, join⟩ do
  if coord≠⊥ then
    add ⟨MGM_k, LEC_k⟩ to groupView
    if coord=i then                                  {MGM_i is the coordinator }
      send ⟨view, groupView⟩ to MGM_k
      add MGM_k to active-tunnels for Δ time units
  else
    if joiners-list=∅ then                           {Not in group}
      set abstinence← true for 4Δ time units
    else
      add MGM_k to joiners-list
```

Figure 5.4: Actions taken upon receiving an MGM join message.

```
processing ⟨MGM_k, view, v⟩ do
  coord ← k
  groupView ← v
  set leaveFlag←false for Δ time units
```

Figure 5.5: Actions taken upon receiving the group's view.

5.4 Leaving a Group

When an MGM intends to leave the group, it sends a *leave* message to the group's members (Figure 5.6 line 7). An MGM can leave the group only if its *coord*$\neq\perp$ and if Δ time units passed after it received the group's view. If an MGM needs to leave the group during an ongoing election procedure, it must wait for the election to end (Figure 5.6 line 2). An MGM receiving a leave message updates its *groupView*, i.e., removes the leaving MGM from its view (Figure 5.7). If the coordinator intends to leave the group, it must appoint another MGM to become the new coordinator. If the group's view is empty, then the coordinator can leave without sending any messages to other MGMs. In the MaGMA overlay solution, we use a different approach to coordinator transition from the one used in MGMLeader above. Instead of requesting permission to leave from the coordinator, non-coordinator MGMs may leave the group at will. The coordinator, on the other hand, needs to ensure that it has a successor before leaving the group. The coordinator elects an MGM, queries this MGM using a *request-transition* and wait for the MGM's reply (Figure 5.6 lines 12–14). If the elected MGM declines the transition the coordinator needs to find a new candidate for the transition. If it grants the transition, i.e., sends a *transition-granted* message, the coordinator establishes a tunnel towards the new coordinator for Δ time units and then leave the group. After granting the transition the new coordinator updates the rest of the MGMs participating in the group regarding the transition by sending them a *new-coord* message (Figure 5.9 line 10). When an MGM receives a *new-coord* message it removes the previous coordinator from its *groupView* and updates its *coord* value with the new coordinator UID (Figure 5.8). This approach expedites leaving the group for all MGMs except the coordinator. In Section 5.6 we further discuss the transition procedure.

```
upon leaving group do
  wait until coord≠⊥
  wait until !abstinence
  wait until leaveFlag
  LEC←LEC+1
  if coord≠ i then
    send ⟨leave, LEC⟩ message to MGMs in groupView
  else
    wait until activeTunnels= ∅
    remove MGM_i from groupView
    while coord= i and ‖ groupView ‖> 1 do
      next-coord← coordSelect(groupView)
      send ⟨transition-request, groupView, LEC⟩ to next-coord
      wait for reply from next-coord
      if receiving transition-denied then
        next-coord←⊥
        process messages in msgBuffer and clear msgBuffer
      else                                        {received transition-granted}
        coord←next-coord
        establish a control tunnel towards coord for Δ time
    LEC←LEC+1
    coord←⊥
    next-coord←⊥
    groupView← ∅
```

Figure 5.6: MGM leave code.

```
processing ⟨MGM_k, leave⟩ msg do
  wait until coord≠⊥
  remove MGM_k from groupView
```

Figure 5.7: MGM actions upon receiving leave message.

processing $\langle MGM_k, \textit{new-coord}\rangle$ *msg* **do**
  if *coord*=$\perp$ **then** {*this can occur during transition*}
    wait till receiving *view* message and process
  remove *coord* from *groupView*
  set *coord*$\leftarrow k$

Figure 5.8: MGM actions upon receiving new-coord message.

processing $\langle MGM_k, \textit{transition-request}, v\rangle$ *msg* **do** {MGM_k *is the coord*}
  LEC$\leftarrow$LEC+1
  if *localView*=$\perp$ **then**
    send $\langle \textit{transition-denied}, LEC\rangle$ to *coord*
  else
    send $\langle \textit{transition-granted}, LEC\rangle$ to *coord*
    groupView$\leftarrow v$
    remove *coord* from *groupView*
    coord$\leftarrow i$
    send $\langle \textit{new-coord}, LEC\rangle$ message to MGMs in *groupView*
    set *leaveFlag*$\leftarrow$false for Δ time units

Figure 5.9: MGM actions upon receiving transition-request message.

5.5 Handling MNs movements

When an MN moves from one domain to another, it performs a WiFi or cellular handoff and receives a new IP address. It then sends a *move* message to the new MGM with its UID and the UID of its former MGM. The MGM adds the MN to its local view and sends a *handoff* message to the former MGM, only if it needs to join the group, i.e., the moving MN is the first MN participating in the group (Figure 5.11). When the former MGM receives such a message (Figure 5.12), it removes the MN from its local view, forwards the *handoff* message to the coordinator. The coordinator, in turn, forwards *join* messages on behalf of the new MGM to the group's MGMs, and sends the current group view to the new MGM. This message flow is illustrated in Figure 5.10.

If the new MGM already participates in the group, it does not send the *handoff* message to the previous MGM. The previous MGM detects the absence of the MN and removes the MN from its *localView*. If the *localView* becomes empty due to the MN's movement the MGM leaves the group as described in Section 5.4.

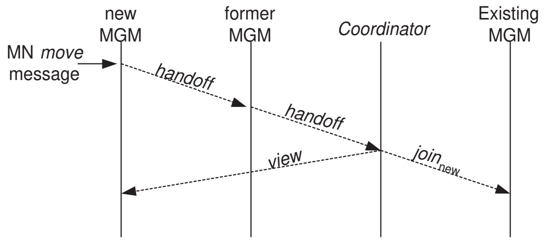


Figure 5.10: Move message flow in MaGMA Multicast Overlay.

upon receiving MN_k move message $\langle move, MGM_l \rangle$ **do**
  wait until !*abstinence*
  LEC←LEC+1
  if *groupView*= $\emptyset$ **then** {*need to join the group*}
    send $\langle i, handoff, LEC, MN_k \rangle$ to MGM_l

Figure 5.11: MGM move code.

5.6 Coordinator Transition

When all the MNs located in the coordinator's domain leave the group or move to another domain, the coordinator will attempt to leave the group. The coordinator can not leave the group immediately after its *localView* becomes empty. It needs to verify that its *active-tunnels* is empty as well, and that its *leaveFlag*= *true*, i.e., the MGM has functioned as the coordinator at least Δ time units. When all three conditions hold, the coordinator can start the transition procedure. It is the coordinator's responsibility to designate another MGM as the new coordinator. The coordinator elects an MGM from its group's view, and sends a query message, *transition-request*, to that MGM. If the MGM is active, i.e., its local view is not empty, then it confirms the transition by sending a *transition-granted* message and informs the rest of the MGMs in the group about the transition by sending them *new-coord* messages. If its local view is empty (this can happen if the MGM has sent a *leave* message during the transition query), then it replies with a *transition-denied* message, and the coordinator needs to query another MGM.

During the transition procedure, all MGM messages received by the coordinator are buffered and forwarded to the new coordinator, as described in Figure 5.2 Lines 6–8. If the coordinator receives a *transition-denied*, then it handles the buffered messages. If it receives a *transition-granted* message, it leaves the group. In addition, it forwards to the new coordinator all MGM messages sent prior to the transition, for up to a Δ time units. To avoid

```
processing ⟨MGM_k, handoff, MN_l⟩ msg do
  wait until !abstinence
  if MN_l ≠⊥ then                                              {MGM_i is the previous MGM}
    remove MN_l from localView
    LEC←LEC+1
  if coord≠ i then
    add MGM_k to groupView
    send ⟨MGM_k, handoff, LEC_k, ⊥⟩ to coord
    if localView= ∅ then
      wait until leaveFlag
      send ⟨leave, LEC⟩ message to MGMs in groupView
  else
    if MGM_k ∉ groupView then
      send MGM_k join message to MGMs in groupView
      add MGM_k to groupView
      add MGM_k to activeTunnels
      send ⟨view, groupView⟩ to MGM_k
    if MN_l ≠⊥ then
      if localView= ∅ then
        wait until activeTunnels≠ ∅
        wait until leaveFlag
        while ∥ groupView ∥> 1 do
          next-coord← coordSelect(groupView)
          send ⟨transition-request, groupView, LEC⟩ to next-coord
          add next-coord to active-tunnels
          wait for reply from next-coord
          if receiving transition-denied then
            remove next-coord from active-tunnels
            process messages in msgBuffer and clear msgBuffer
            next-coord←⊥
          else                                                 {received transition-granted}
            coord←next-coord
            establish a control tunnel towards coord for Δ time
      LEC←LEC+1
      coord←⊥
      next-coord←⊥
      groupView← ∅
```

Figure 5.12: MGM handoff code.

multiple tunneling, which would cause extra delay in message processing, a coordinator can initiate a new transition only after it functions as a coordinator at least Δ time units.

If the coordinator's local view and group view are empty, i.e., no MNs and MGMs are participating in group, then it can leave the group without sending any transition message. In order to avoid frequent coordinator changes and election procedures that flood the network, the coordinator needs to wait a minimum time interval before electing a new coordinator or leaving the group.

We now explain the selection of the 4Δ time units guard time in the election procedure. When a new MGM sends a *join* message for an active group, the group's coordinator may be in transition. In this situation, the coordinator does not reply immediately. If the transition succeeds, then the new coordinator replies with the group's view, which takes a total of at most 3Δ time units. On the other hand, if the transition fails, i.e., the coordinator receives a *transition-denied* message, then it can take up to 3Δ time units for the coordinator to handle the buffered message, and up to 4Δ time units for the joiner to get a response. These scenarios are depicted in Figure 5.13.

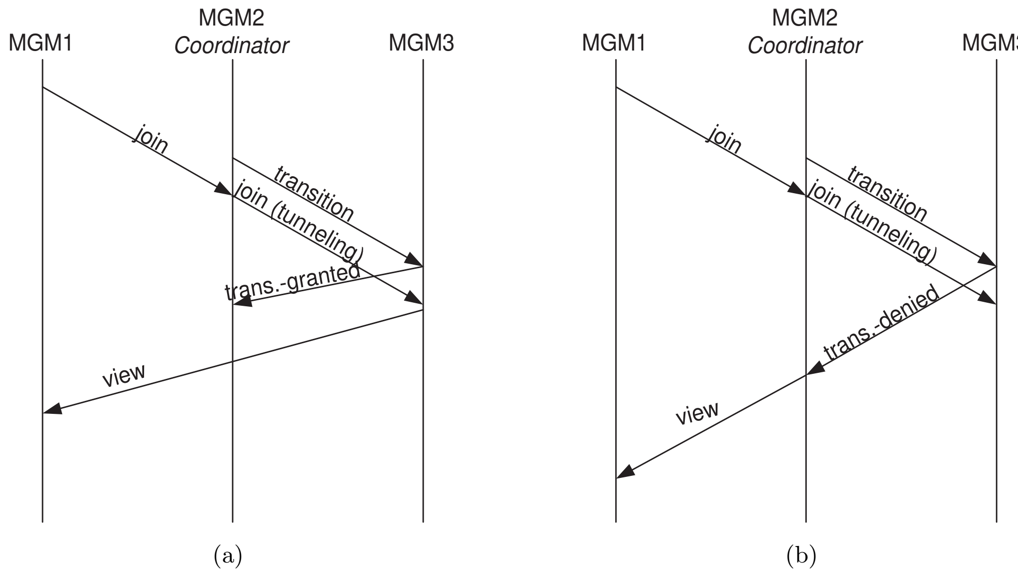


Figure 5.13: Two coordinator transition scenarios, where the group view is received within (a) 3Δ and (b) 4Δ time units.

According to this calculation, in MaGMA Multicast Overlay $T_{respond} = 3\Delta$. As $T_{guard} = 4\Delta$, the delay of the scenario depicted in Figure 4.4 Section 4.2.3, bounds MN's movement by 10Δ time units, as required in Section 3.1.

Again, as mentioned in Section 5.3, it is possible to use the improved join method, which effects the transition procedure, as leaving coordinator needs to inform not only the newly elected coordinator and the participating MGMs but also the representing MGM.

5.7 Ensuring View Consistency

The main goal of the control plane is keeping view consistency. The control plane must enable the MGMs to hold a coherent and up-to-date view of the MGMs participating in the group, as well as maintain an accurate view of the local MNs participating in the group. In Appendix C we provide a formal proof that the protocol keeps view consistency, here we intuitively describe several possible problems and their solutions.

Due to the fact that MGMs can join and leave the group, a potential inconsistency can occur with careless handling of joins, when two or more MGMs join the group simultaneously. Figure 5.14(a) illustrates a scenario that could have occurred if we would not have used tunnels. In this example, MN2 joins the group and MN1 moves from one domain to another. Neither MGM1 nor MGM2 are in the group before these events. Due to MN2's *join* message MGM2 broadcasts a *join* message. This message arrives to MGM1 before it receives MN1's *move* message, thus MGM1 ignores MGM2's message. Upon receiving MN1's message, MGM1 sends a *handoff* message to the former MGM. The message is forwarded to the coordinator, which receives MGM1's message before MGM2's message. Thus, MGM1 is unaware of MGM2 and holds an inconsistent view of the group.

MaGMA's solution for this problem is to tunnel new events from the coordinator to newly added group members. The group coordinator receiving a message (*join* or *move*) from a new MGM forwards to that MGM all incoming messages from other MGMs for Δ time units. Thus, new MGMs receive all group's events and maintain a coherent and up-to-date view of the group. The solution is illustrated in Figure 5.14(b).

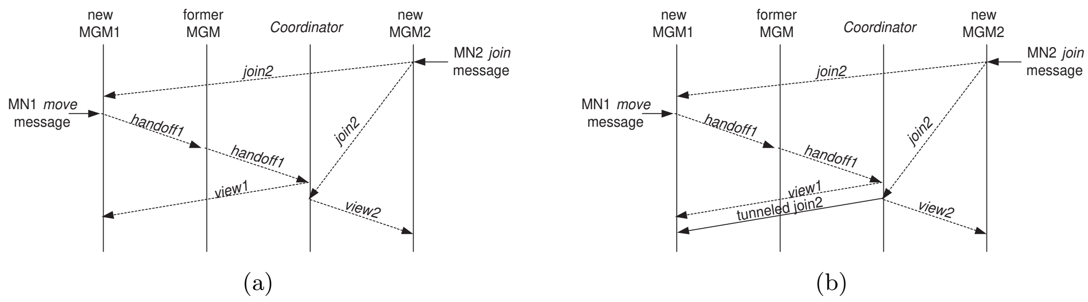
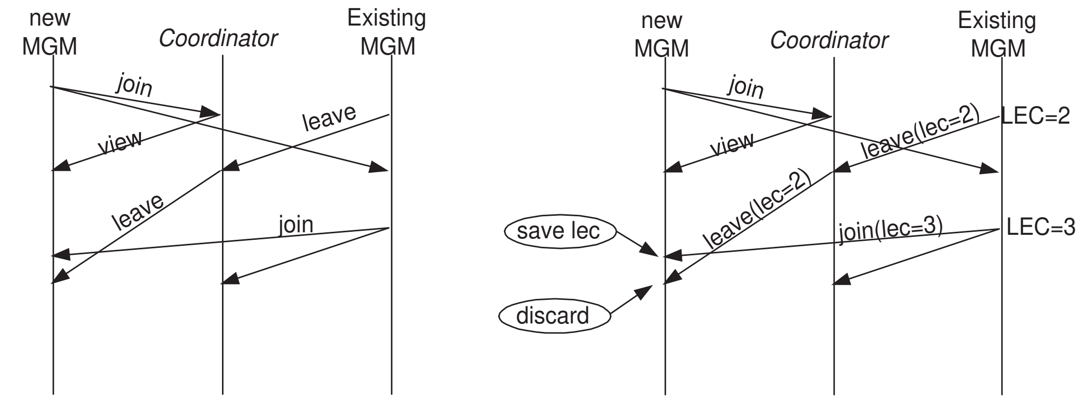


Figure 5.14: (a) An examples for potential view inconsistency with a simplified control plane and (b) the tunneling solution.

Due to the role of the coordinator as the forwarder of some but not all of the MGMs' messages, in some cases, an MGM can receive the message of another MGM out of order. For example in Figure 5.15(a), a new MGM is joining the group and broadcasts a *join* message to the MGMs in the network. The coordinator sends to the new MGM the group's view. Meanwhile, an existing MGM sends a *leave* message. This message is sent prior to the reception of the *join* message. Upon receiving the *leave* message, the coordinator tunnels the message to the new MGM. Meanwhile, the leaving MGM re-joins the group by broadcasting a *join* message. The *join* message is received by the new MGM before the reception of the previous *leave* message. If this situation is not adequately handled, as a result the new MGM may hold an inconsistent view of the group, due to the fact that it removed the existing MGM from its *groupView*. To address this difficulty, each MGM maintains an increasing *Local Event Counter* (LEC) for every group, initialized to 1. Whenever an MGM receives a *join*, *leave*, or *move* message from a local MN, it increments the appropriate LEC. In addition, the MGM keeps, for every group, a *LECvector*, holding the highest known LEC for each MGM in this group. The LEC solution for the scenario described above is illustrated in Figure 5.15(b).

In every message sent from one MGM to another, the sender's LEC is included. When an MGM receives a packet, it checks the LEC. If the LEC in the message is lower than the known one, then it ignores the message. When the coordinator sends the group's view to new joining MGMs, it includes the LECs corresponding to the view it is sending to the new MGM.



(a) (b)

Figure 5.15: (a) An example for potential view inconsistency with a simplified control plane and (b) the LEC solution.

We now describe another scenario, illustrated in Figure 5.16(a), where out-of-order reception of messages might have caused view inconsistency. In this scenario, MGM1 and MGM2 join a group. The group's coordinator receives MGM2's *join* message before receiving MGM1's *join* message. Therefore, when it receives MGM1's *join* message, it sends back to MGM1 the group's view and tunnels the *join* message to MGM2. MGM1 receives the group's view almost immediately and then it leaves the group by sending *leave* message to MGMs in its *groupView* (the coordinator and MGM2). MGM2 receives the message immediately and therefore removes MGM1 from its *groupView*. Right after that, MGM2 receives the tunneled *join* message previously sent by MGM1, and adds MGM1 to the group's view. In addition, MGM1's *leave* message is received by the coordinator after it closes the tunnel towards MGM2, and therefore the coordinator does not forward the message to MGM2. Consequently, MGM2 thinks that MGM1 is a member of the group. We note that the LEC mechanism cannot prevent this inconsistency as MGM2 removed MGM1 from its *groupView* and therefore does not have any information regarding MGM1. MaGMA's solution to this problem is a guard time maintained after receiving the group's view, requiring that an MGM can not leave the group less than Δ time units after receiving the group's view from the coordinator, as illustrated in Figure 5.16(b) (see Figure 5.6 line 4 and Figure 5.5 line 4). This way, we prevent out-of-order reception of consecutive *join* and *leave* messages.

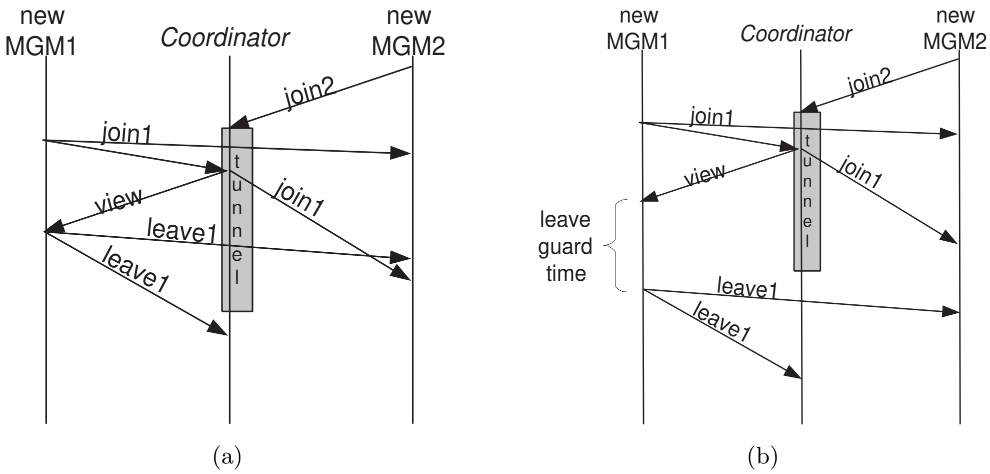


Figure 5.16: (a) An example for potential view inconsistency with a simplified control plane and (b) the guard-time solution.

Chapter 6

Control Plane Evaluation

6.1 Simulations and Analysis Scenario

We now evaluate the overhead associated with the control protocol. We simulate the following uniform network model:

- 10 domains (Domains 1-10), 1 MGM in each domain;
- 10-100 receiving MNs, initially uniformly distributed in Domains 1-10, then moving among these domains;
- a single group, where every receiving MN participates in this group;
- Constant delay between MGMs, $\Delta = 0.4sec$.

6.2 Comparing MaGMA's Two Solutions

As described in Section 3.2, we assume that *move* messages dominate the control traffic. Therefore we simulate MaGMA's control protocols in this setting, and measure the average control overhead associated with a single *move* message in both models. The average is calculated over 1000 events for each number of MNs. In each event, a random MN moves to a new random domain. The results are depicted in Figure 6.1, with 95% confidence intervals for MaGMA Multicast Overlay. We also mathematically analyze the expected control overhead. The detailed analyses of MGMLeader and the MaGMA multicast overlay solution are presented in Appendix A and in Section 6.3.1, respectively. For MGMFlood, this is straightforward. Since each control message is sent to all MGMs, and there are nine receiving MGMs, together with the MN's *move* message the overhead is exactly ten messages per *move* event. Not surprisingly, the analysis and simulation results for this protocol accurately match each other (see Figure 6.1).

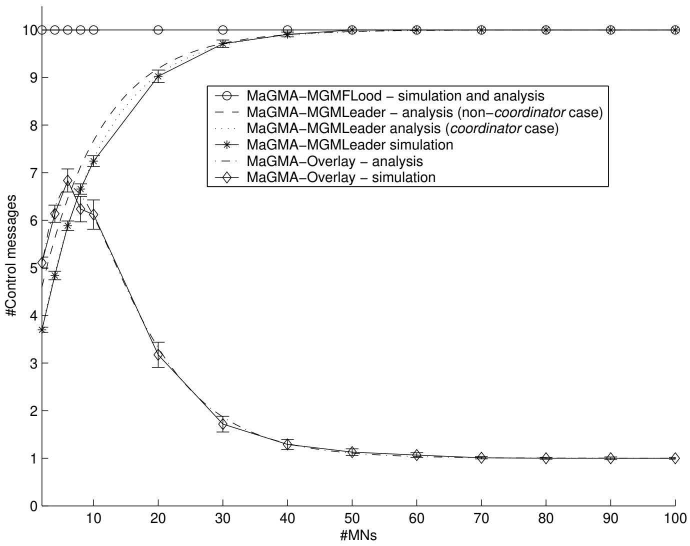


Figure 6.1: Average number of control messages per movement, uniform system, varying number of MNs, fixed number of domains: analysis vs. simulations.

For MGMLeader and the MaGMA overlay solution, we give two separate analyses covering two different situations. Recall that in these protocols, a new MGM joining a group communicates with the MN's former MGM, which forwards the message to the coordinator. We distinguish the case that the former MGM is the group's coordinator (the *coordinator case*), from the case that the former MGM is not the coordinator (the *non-coordinator case*). The analyses of these cases are depicted separately in Figure 6.1. In MGMLeader, one more control message is sent in the non-coordinator case as compared to the coordinator case—the message from the former MGM to the coordinator.

In MaGMA's multicast overlay solution, the situation is reversed: if the group becomes empty following the move, then in the non-coordinator case, the former MGM can leave the group without asking for permission, whereas in the coordinator case, the coordinator needs to find a new coordinator to replace it and to inform other MGMs of the transition. Therefore, the overhead in the coordinator case is larger.

With MGMLeader, the overhead increases with the number of MGMs that have members in the group. In sparse groups, few MGMs are involved, and hence few control messages are sent. With MaGMA's multicast overlay solution, the overhead is similar to MGMLeader in sparse groups (2-10 MNs), but then the overhead *decreases* with the number of MGMs that have members in the group. In dense groups (50-100 MNs), most movements do not cause any changes in MGM-level membership, and therefore MaGMA's overlay solution sends only two messages per movement: from the moving MN to the new MGM and from the new MGM to the former MGM. The remaining MGMs do not need to be informed of the move.

We conclude that in the subscription model, MGMLeader is preferable for sparse groups, whereas the much simpler MGMFlood may be adequate for dense groups in which all or most MGMs participate. It is also clear that for large groups, MaGMA's multicast overlay greatly outperforms MGMFlood and MGMLeader, and this solution is therefore preferable where MGMs that support this functionality can be deployed.

6.3 Evaluating MaGMA Multicast Overlay

6.3.1 Control Overhead

We now evaluate the control overhead of join, leave, movement and disconnection of an MN in the MaGMA multicast overlay model. The average number of control messages sent per event are depicted in Figure 6.2.

We denote N as the number of MNs in the network and D as the number of domains in the network. We assume that MNs are uniformly distributed among domains. There is a single active group therefore every MGM that has MNs within its domain participates in the group.

Analysis of move event

We now analyze the average control traffic overhead in the MaGMA Overlay protocol. When an MN moves from one domain to another there are four possible cases:

1. The new domain has no MNs and the previous domain becomes empty (the previous domain had a single MN, the MN that moved).
2. The new domain has no MNs and the previous domain is not empty.
3. The new domain has MNs and the previous domain becomes empty.
4. The new domain has MNs and the previous is not empty.

We note that case 4 is a special case where the only messages sent are from the moving MN to the new MGM.

We now calculate the probability of each case:

1. $p_1 = \left(1 - \frac{2}{D}\right)^{N-1}$
2. $p_2 = \left(1 - \frac{1}{D}\right)^{N-1} - \left(1 - \frac{2}{D}\right)^{N-1}$
3. $p_3 = \left(1 - \frac{1}{D}\right)^{N-1} - \left(1 - \frac{2}{D}\right)^{N-1}$
4. $p_4 = 1 - 2\left(1 - \frac{1}{D}\right)^{N-1} + \left(1 - \frac{2}{D}\right)^{N-1}$

In each of the aforementioned cases messages are sent to a subset of the MGMs (MGMs that participate in the group). We now calculate the probability that a domain (that is not the new or previous domain) participates in the group, i.e., has MNs that participate in the group:

1. $p_{1,dne} = \left(1 - \left(1 - \frac{1}{D-2}\right)^{N-1}\right)$
2. $p_{2,dne} = \frac{1}{p_2}\sum_{i=2}^{N}\left[\binom{N-1}{i-1}\left(\frac{1}{D}\right)^i\left(1 - \frac{2}{D}\right)^{N-i}\left(1 - \left(1 - \frac{1}{D-2}\right)^{N-i}\right)\right]$
3. $p_{3,dne} = p_{2,dne}$

We now calculate the average number of MGMs participating in the group, beside the new and previous MGM. We use an indication function, denoted as δ_i

$$\delta_i = \begin{cases} 0 & \text{if MGM}_i \text{ does not participate in group,} \\ 1 & \text{if MGM}_i \text{ participates in group.} \end{cases} \tag{6.1}$$

Therefore the average number of MGMs participating in each case is

$$E_i = \sum_{\substack{j=1 \\ j\neq \text{prev} \\ j\neq \text{new}}}^{D} \delta_j p_{i,dne} = (D-2)p_{i,dne} \qquad i = 1,2,3 \tag{6.2}$$

We examine two possible scenarios: the previous MGM is the group's coordinator and the previous MGM is a regular MGM. For each of these scenarios and for each of the cases we calculate the number of messages sent:

- Non coordinator scenario

  1. The MN sends a *move* message to the new MGM, the new MGM sends a *handoff* message to the previous MGM. The previous MGM forwards the message to the group's coordinator, 3 messages. In addition the previous MGM sends a *leave* message to the MGMs in the group, E_1 messages. Upon receiving the *handoff* message, the coordinator sends the group's view to the new MGM and a *join* message to the rest of the MGMs in the group, $1 + (E_1 - 1)$ messages. Thus, the total messages sent in this scenario $2E_1 + 3$.

2. The MN sends a *move* message to the new MGM, the new MGM sends a *handoff* message to the previous MGM. The previous MGM forwards the message to the group's coordinator, 3 messages. The coordinator sends the group's view to the new MGM and a *join* message to the rest of the MGMs in the group, $1+(E_2-1)$ messages. Thus, the total messages sent in this scenario E_2+3.
3. The MN sends a *move* message to the new MGM, the new MGM already participates in the group, thus it does not inform the previous MGM. The previous MGM detects that the MN is not in its domain and sends a *leave* message to the MGMs participating in the group. Total of E_3+2 messages.
4. In this scenario the new MGM already participates in the group and the previous MGM does not need to leave the group. Therefore, the only message sent in the *move* message: total of 1 message.

- coordinator scenario

1. The MN sends a *move* message to the new MGM, the new MGM sends a *handoff* message to the previous MGM (the coordinator). The coordinator sends a *join* message to the MGMs in the group, E_1 messages, and a *view* message to the new MGM. In addition it initiates a transition procedure where it sends a *transition-request* message and receives a *transition-granted* message, 2 messages. The new coordinator sends a *new-coord* message to the rest of the MGMs in the group, E_1 messages. Therefore, the total number of messages sent is $2E_1+5$.
2. The MN sends a *move* message to the new MGM, the new MGM sends a *handoff* message to the previous MGM (the coordinator), 2 messages. The coordinator sends the group's view to the new MGM and a *join* message to the rest of the MGMs in the group, $1+E_2$ messages. Thus, the total messages sent in this scenario E_2+3.
3. The MN sends a *move* message to the new MGM, the new MGM already participates in the group, thus it does not inform the previous MGM. The previous MGM (the coordinator) detects that the MN is not in its domain and it initiates a transition procedure. It sends a *transition-request* to an MGM from its *groupView*, the MGM sends back a *transition-granted* message and sends to the rest of the MGMs in the group *new-coord* message, $2+E_3$ messages. Therefore, the total number of messages sent is E_3+3 messages.
4. Similar to the non-coordinator case, in this scenario only 1 message is sent.

We now calculate the probabilities for a coordinator scenario in cases 1 and 3:

- In scenario-1 the MN moves from the coordinator's domain into an empty domain, therefore if the MN is located in domain i $(i \neq 1, D)$ it can move to domain $i-1$ or to domain $i+1$. If the MN is located in domain 1 it can move only to domain 2 and if the MN is in domain $D-1$ it can move only to domain $D-2$.We note that the MN can not move from domain D, and that domains $1, ..., i-1$ must be empty. In addition we note the moving MN is the only MN located in the coordinator's domain.
  - MN in domain-1 moves to domain-2: $\frac{1}{2}\frac{1}{D}(\frac{D-2}{D})^{N-1}$.
  - MN in domain-2 moves to domain-1 or to domain-3: $\frac{1}{2}\frac{1}{D}(\frac{D-2}{D})^{N-1} + \frac{1}{2}\frac{1}{D}(\frac{D-3}{D})^{N-1}$
  - MN in domain-i $(i \neq 1, D-1, D)$: $\frac{1}{2}\frac{1}{D}(\frac{D-i}{D})^{N-1} + \frac{1}{2}\frac{1}{D}(\frac{D-(i+1)}{D})^{N-1}$
  - MN in domain-$D-1$: $\frac{1}{2}\frac{1}{D}(\frac{D-(D-1)}{D})^{N-1}$.
- In scenario-3 the MN moves from the coordinator's domain into a non-empty domain, therefore the moving MN can be located only in domains $1, ..., D-1$ and moves only towards the domain on its right. In addition we note the moving MN is the only MN located in the coordinator's domain and that there is at least one MN in the new domain.
  - MN in domain-1 moves to domain-2: $\frac{1}{2}\frac{1}{D}\frac{1}{D}(\frac{D-1}{D})^{N-2}$.
  - MN in domain-$i = 1, ..., D-1$: $\frac{1}{2}\frac{1}{D}\frac{1}{D}(\frac{D-(i+1)}{D})^{N-2}$.

Therefore, the probability for a coordinator case in scenario 1 is

$$p_{1,coord} = \sum_{i=1}^{D-2} \frac{1}{2}\frac{1}{D}\left(\frac{D-(i+1)}{D}\right)^{N-1} + \sum_{i=2}^{D-1} \frac{1}{2}\frac{1}{D}\left(\frac{D-i}{D}\right)^{N-1} \tag{6.3}$$

And in scenario 3 is

$$p_{3,coord} = \sum_{i=1}^{D-1} \frac{1}{2}\left(\frac{1}{D}\right)^{2}\left(\frac{D-i}{D}\right)^{N-2} \tag{6.4}$$

Therefore the average number of messages sent is

$$\begin{aligned}\overline{N}_{move} =& p_1\left(p_{1,coord}\left(2E_1+4\right) + \left(1-p_{1,coord}\right)\left(2E_1+3\right)\right)\\ &+ p_2\left(E_2+3\right)\\ &+ p_3\left(p_{3,coord}\left(E_3+2\right) + \left(1-p_{3,coord}\right)\left(E_3+1\right)\right)\\ &+ p_4\end{aligned} \tag{6.5}$$

Analysis of join event

We now analyze the average control traffic overhead in the MaGMA Overlay protocol when a new MN joins the group. The new joining MN uniformly chooses a domain from the D available domains.

The joining MN sends a *join* message to the MGM. If the domain is empty the MGM sends a *join* message to the rest of the MGMs, $D-1$ MGMs. On the other hand if the MGM already participates in the group, i.e., there are MNs in the domain, the MGM does not need to send any message. The probability that the domain is empty

$$p_{empty} = \left(1 - \frac{1}{D}\right)^N \tag{6.6}$$

Thus, the average number of control messages sent per join event

$$\overline{N}_{join} = (D-1)\, p_{empty} + 1 \tag{6.7}$$

Analysis of leave event

We now analyze the average control traffic overhead in the MaGMA Overlay protocol when a new MN joins the group. The leaving MN is uniformly choosen from the N available MNs. The probability that an MN is located in a domain, denoted as p_{sel} is:

$$p_{sel} = \frac{1}{D} \tag{6.8}$$

Thus the probability that an MGM needs to leave the group, equals to

$$p_{leave} = \left(\frac{D-1}{D}\right)^{N-1} \tag{6.9}$$

The probability that the leaving MGM is the group's coord, denoted as p_{coord}, depends on the ID of the MGM. It is clear that according to the election procedure the MGM with the lowest ID is elected to become the group's coordinator. Therefore, the MGM has to be the MGM with the lowest ID among the participating MGMs. In addition, there are only $D-1$ options to the coordinator, this stems from the fact that if the coordinator is MGM D then all the MNs are located in its domain and it does not need to leave the group. Therefore

$$p_{coord} = \sum_{i=1}^{D-1} \frac{1}{D} \left(\frac{D-i}{D}\right)^{N-1} \tag{6.10}$$

When the a regular MGM leaves the group it sends a *leave* message to the MGMs in the group. The average number of MGMs in a group (beside the leaving MGM) is

$$E = (D-1)(1-(1-p_{sel})^{N-1}) \tag{6.11}$$

If the coordinator leave the group, it sends a *transition-request* to the new coordinator, the new coordinator sends a *transition-grant* message to the coordinator and a *new-coord* message to the rest of the MGMs in the group, total of $2 + E - 1$ messages. Thus the average number of messages sent upon an MN's leave event is

$$\overline{N}_{leave} = 1 + p_{leave}(p_{coord}(E + 1) + (1 - p_{coord})E) \tag{6.12}$$

Analysis of disconnect event

The only difference between leave and disconnect event is the message sent by the MN. The disconnected MN is uniformly choosen from the N available MNs. Thus, based on Equation 6.12, the average number of messages sent per a disconnect event

$$\overline{N}_{disconnect} = \overline{N}_{leave} - 1 \tag{6.13}$$

6.3.2 Time to Stabilization

Another performance parameter of MaGMA that we examine is time-to-stabilization (TTS). This parameter measures the amount of time it takes an event to propagate through the network/group. For example, when an MN joins or leaves the group or when an MN moves to another domain, it sends a message to the local MGM indicating this event. As a result, further messages are sent by the local MGM and other MGMs. TTS measures the time from the beginning of the event until the time where the last message sent due to this event is received. We consider two measurements of TTS, one from the MN's perspective denoted as TTS_{MN}, where we measure TTS from the time where the MN sends a message or from the time of its disconnection, and the second from the MGM's perspective, denoted as TTS_{MGM}, where we measure TTS from the time the MGM receives the MN's message or detects the MN's disconnection.

Table 6.1 summarizes the maximum possible TTS_{MN} and TTS_{MGM} of the four events. The maximal TTS for a join event occurs when the MN sends a *join* message to the local MGM and the MGM needs to join the group, for a leave event, the maximal TTS occurs when the leaving MN's MGM functions as the coordinator and need to initiate a transition. Similarly, for the move and disconnect events the maximal TTS is receives when there is a transition procedure due to the MN's event.

We evaluate the average TTS of the four events using simulations based on the uniform network. Figure 6.3 shows the results; 95% confidence intervals are shown. The results of the TTS_{MGM} are depicted in Figure 6.3(a) and the results of the TTS_{MN} are depicted in Figure 6.3(b).

	max TTS_{MN}	max TTS_{MGM}
join	3Δ	2Δ
leave	3Δ	2Δ
move	$\tau + 3\Delta$	3Δ
disconnect	$\tau + 2\Delta$	2Δ

Table 6.1: Maximum TTS according to the four events.

From Figure 6.3(a) it is clear that when the network becomes denser the TTS_{MGM} is equal to zero for all events, this stems from the fact that MGMs do not need to send further messages upon an event, for example if an MGM receives a *join* message it does not need to join the group as it already participates in the group. In sparse networks, we note that for move events the TTS is equal to the maximal TTS, see Table 6.1, and that the TTS of leave and disconnect events is similar, this stems from the fact that MGMs do not distinguish between these events.

From Figure 6.3(b) we see that the average TTS of the move events decreases with the number of the MNs, and in dense network (more than 20 MNs) the TTS is around $\tau = 1.1sec$, similar to the disconnect event. The main difference between TTS_{MGM} and TTS_{MN} is with disconnect and move events, due to the fact that detection time is larger than the delay of a message, and that join and leave events are based on message passing.

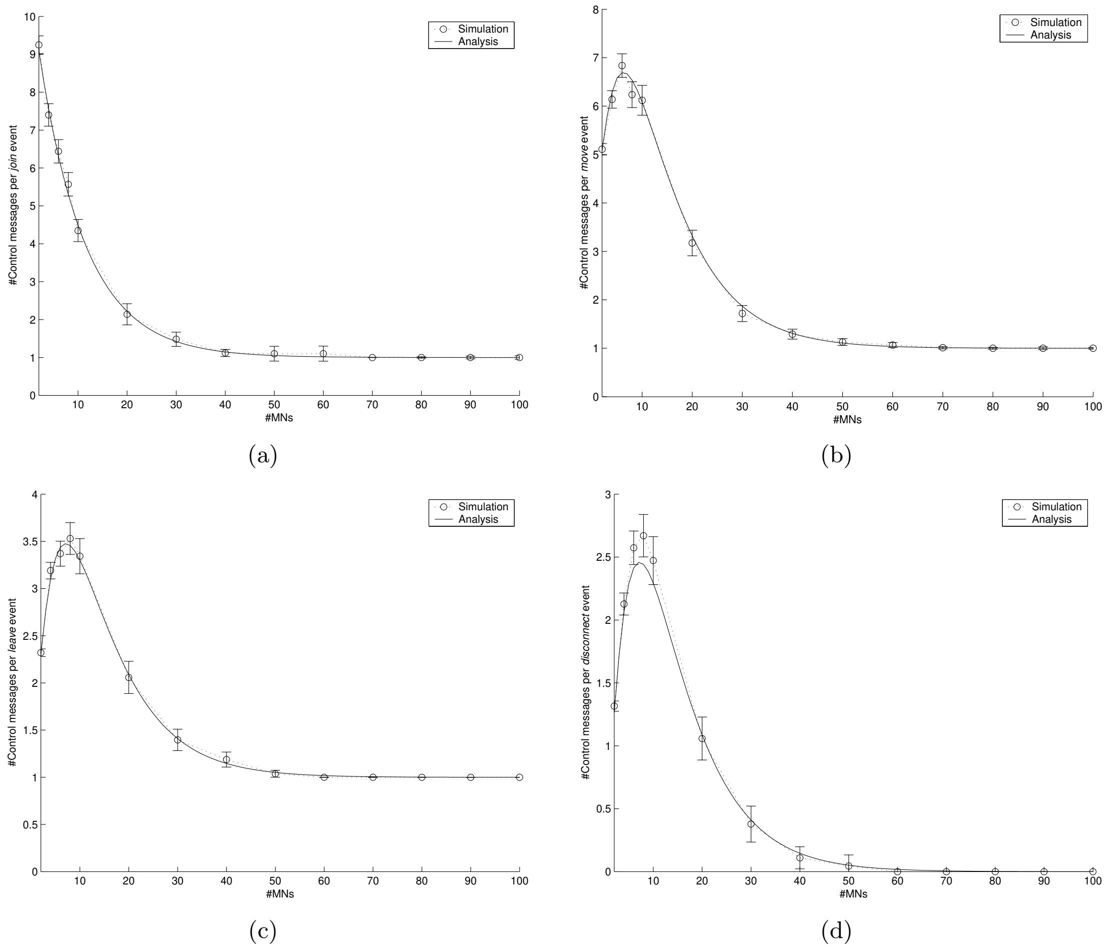
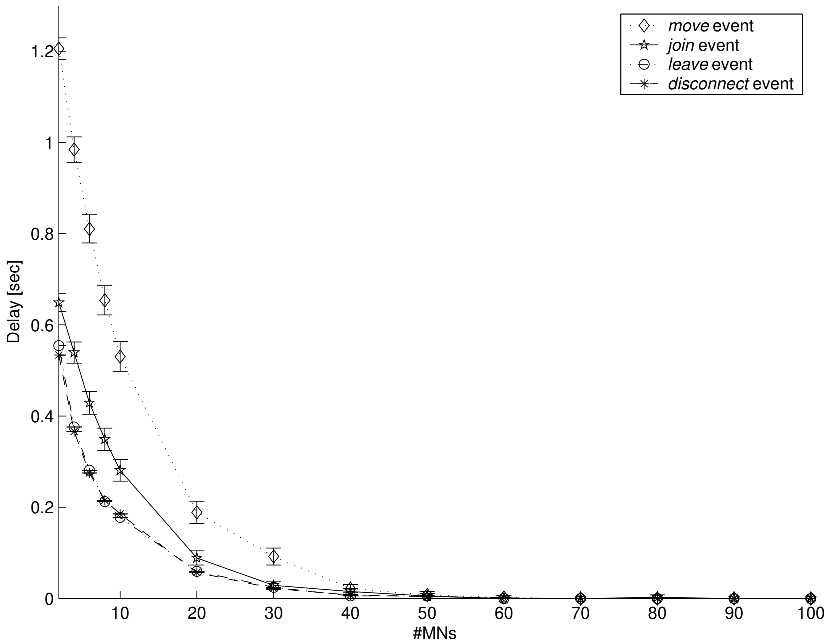
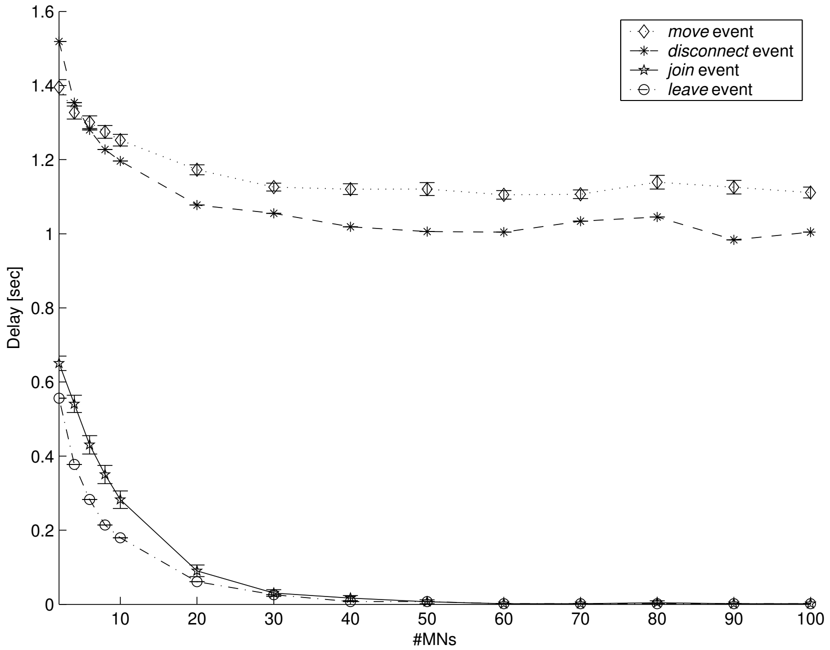


Figure 6.2: Average number of control messages per (a) join, (b) movement, (c) leave and (d) disconnect event, uniform system, varying number of MNs, fixed number of domains: analysis vs. simulations.



(a)



(b)

Figure 6.3: Time to stabilization per event (a) from MGM detection (b) from the beggining of the event, $\Delta = 0.4sec$, $\tau = 1.1sec$, uniform system, varying number of MNs, fixed number of domains: simulation.

Chapter 7

MaGMA's Data Plane

Until this point, we have focused on MaGMA's control plane. We now examine the data plane, i.e., the transport of multicast data among group members.

7.1 Comparing MaGMA's Two Solutions

We now compare the data planes of the two MaGMA solutions. In the subscription model, a sender wishing to initiate a multicast session retrieves the group view and creates unicast streams to all the group members in order to send data to them. In the multicast overlay solution, the sender sends data messages to its MGM indicating the target group, and the MGM forwards the messages via the overlay to other MGMs listed in its group view. Upon receiving data messages, the MGMs forward the messages to their local MNs participating in the group (using multicast or unicast communication within that domain). The latter approach reduces the load on the IP infrastructure and supports larger groups better.

To illustrate the differences between the two models, we analyze the average number of incoming streams per domain. The detailed analysis is available in Appendix B. We use the following uniform network in our calculations:

- 11 domains (Domains 0-11), 1 MGM in each domain;
- 10-200 receiving MNs, uniformly distributed in Domains 1-10;
- 8 groups, where every receiving MN participates in a single group chosen uniformly at random;
- a fixed number of sources in Domain 0;
- all groups are active.

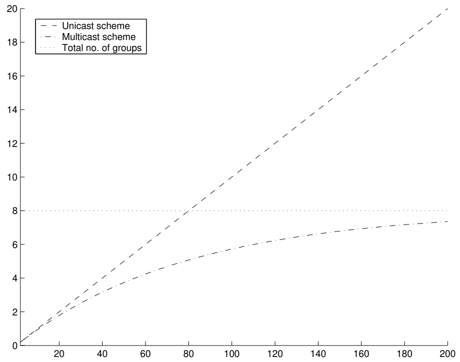


Figure 7.1: Average number of incoming streams per domain in uniform system with 8 groups, varying number of MNs, fixed number of domains: analysis.

Figure 7.1 shows that using the unicast scheme, the average number of incoming streams is the average number of participating MNs within the domain, whereas using the multicast scheme, it is the average number of groups in the domain.

7.2 Comparison with Mobile IP

We now examine the data plane of MaGMA and compare it to Mobile IP. The main reason for selecting this kind of comparison is the fact that current solutions, such as PTT in cellular networks, use the same delivery concept as Mobile IP. Traffic is routed through a designated server and only then delivered to the user. The designated server is an analogy to the HA in Mobile IP.

The use of Mobile IP to forward traffic to a mobile user traversing the network suffers from poor performance due to triangle routing. In contrast, in MaGMA, traffic is sent either directly to the destination nodes (in the subscription model) or using the MGM's overlay. In the latter case, if MGMs are located in every domain, or at least in strategic central points in the core, then network traffic is also sent using an almost direct route.

To illustrate this advantage of MaGMA, we simulate a constant bit rate (CBR) UDP session from a static source to a moving receiver. We measure the performance of both MaGMA and Mobile IP. We simulate a network with four domains. The source is located in Domain 0, the receiver is initially located in Domain 1 and then moves toward Domain 3 through Domain 2. Domain 1 functions as the home domain of the receiver in the Mobile IP simulations. Figure 7.2 illustrates the simulated network.

We measure the average end to end delay in three architectures: Mobile IP, MaGMA multicast overlay model, and MaGMA subscription model. Figure 7.3 shows the result of the simulations. It is clear that when the receiver is located at its home domain, the three architectures perform similarly. When the receiver moves outside its home domain, the triangle routing causes the packet delay to increase by a factor of 3, whereas using MaGMA the delay does not increase due to the use of the optimal route.

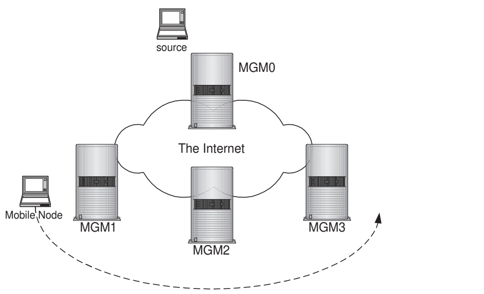


Figure 7.2: Scenario simulated in Figure 7.3.

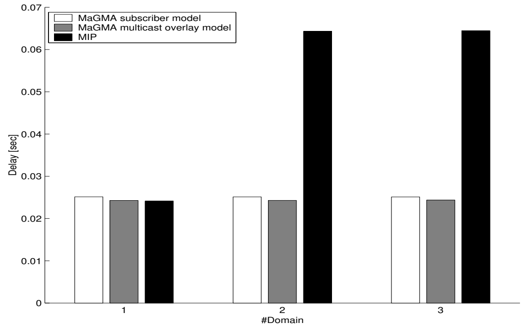


Figure 7.3: Average end-to-end packet delay for system depicted in Figure 7.2: simulation.

Another simulation that we conducted, emphasizes MaGMA's advantages in packet loss comparing to MIP. We measured the packet loss of a CBR UDP session where the source is static and the receiver is an MN. The source is located in Domain 1, the receiver is initially located in Domain 1 and the moves towards Domain 4 through Domains 2 and 3. Domain 1 functions as the home domain of the receiver in the MIP simulations. Figure 7.4 illustrates the simulated network.

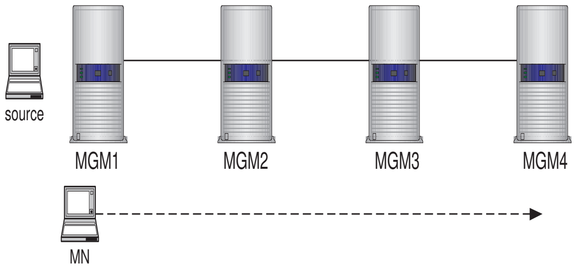


Figure 7.4: Scenario simulated in Figure 7.5.

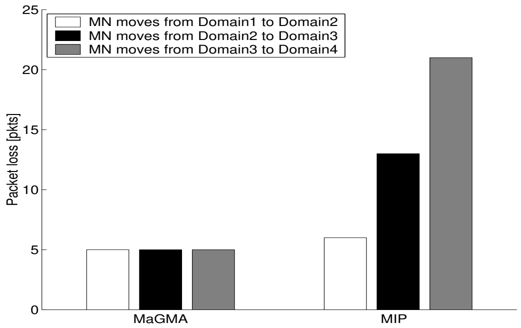


Figure 7.5: Packet loss for system depicted in Figure 7.4: simulation.

We measure the packet loss during the movement of the receiver between domains.

Figure 7.5 shows the result of the simulation. The results emphasize the advantages of MaGMA, while in MIP the packet loss depends on the distance of the MN from its home domain, in MaGMA the packet loss depends on delay between the new and previous MGMs. In MIP, when an MN moves into a foreign domain, it informs the HA regarding its new address. After receiving this update, the HA can tunnel the data messages to the current location of the MN, therefore the delay of the update message, i.e., the distance of the MN from its home, is critical to the number of packet loss. On the other hand, in MaGMA, when a MN moves into a new domain the new MGM establishes a connection with the previous MGM, thus it enables the delivery of data messages prior to the ending of the joining procedure.

Another important parameter is the overhead of the headers added to every data packet. In a simple unicast session, the header is composed of the UDP header, 8 bytes, and the IP address, 20 bytes. In MIP, where the MN is located in its home domain and there is no need to tunnel packets, the overhead is the same as in the simple unicast session. When the MN moves into a foreign domain the HA tunnels the packets to the FA using IP encapsulation, thus 20 extra bytes are added to the header. On the other hand, in MaGMA subscription model, no matter where the MN is located, the overhead is as in the simple unicast session. This is due to the fact that the communication between MNs is conducted using direct unicat sessions.

In the multicast related protocols we compare the multicast extension to MIP with MaGMA multicast overlay model. In MIP with multicast, the HA forwards the multicast messages to MNs located in foreign domains. It uses the IP-in-IP encapsulation, two additional IP headers are added by the HA: one to tunnel the message to the FA, and another to tunnel the message from the FA to the MN. In MaGMA multicast overlay model, the data is forwarded over the overlay. MGMs receiving the data messages and forward the messages to local MNs need to know the group ID. Thus, 4 extra bytes are used to identify the group, as in class D addresses. Figure 7.6 illustrates the headers in each of the aforementioned methods.

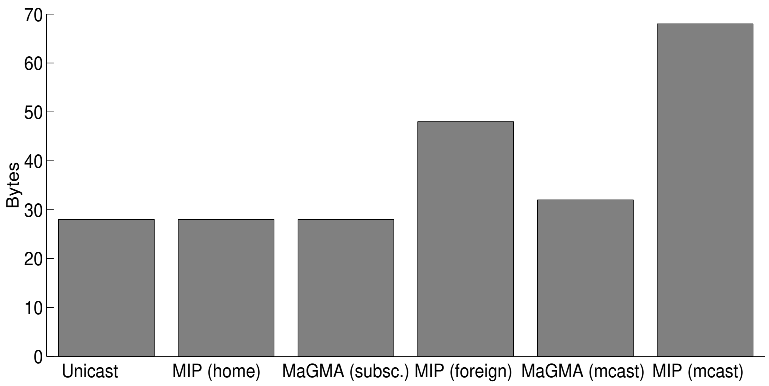


Figure 7.6: Header sizes in unicat and multicast communication methods.

Chapter 8

Prototype Implementation

As a proof-of-concept, we have built a SIP-based prototype of MaGMA's MGM. The MGM uses the NIST-SIP [22] implementation. It is implemented in JAVA, and runs on a standard Intel Pentium 4 PC. Our MNs are WiFi enabled iPAQ PDAs. The MaGMA client software is implemented in C++ on top of the Microsoft Real Time Communication (RTC) client [17]. We have demonstrated MaGMA by running an instant messaging application on the iPAQs.

We have implemented MaGMA with the multicast overlay model, where the MGM serves as a message duplicator. The MGM consists of three components. One component is a SIP parser for processing the received SIP messages. The second is a database that holds information about the current groups and the MNs subscribed to each group. The third part is an MaGMA processor, which updates the database according to the received *move, join, leave* messages, and duplicates and forwards data messages to the group members. The implementation's architecture is illustrated in Figure 8.1.

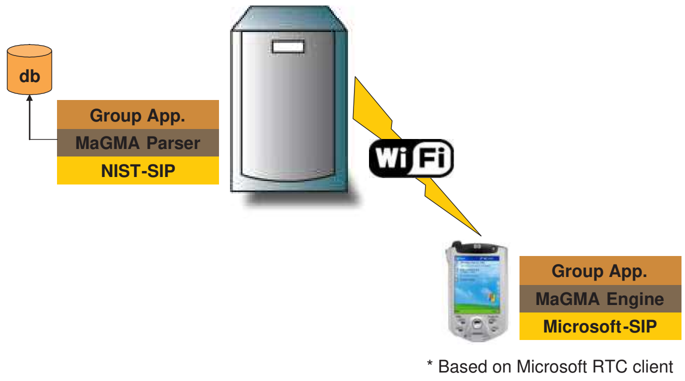


Figure 8.1: MaGMA's implementation.

We illustrate MaGMA's flexibility and general applicability by using a standard wireless environment composed of WiFi access points, regular PC-based servers, and iPAQs, and by basing our prototype on an existing SIP implementation. This work is ongoing and we plan to extend it to larger scale infrastructures.

Chapter 9

Future Work

MaGMA is a general, open, and flexible architecture. We anticipate that the MaGMA-like architectures and similar services will be major components in 3G and B3G networks. As such, we believe that there is plenty of room for proposing enhancements, optimizations, and service extensions beyond the basic services and protocol suite introduced in this paper. We now briefly outline possible service extensions.

Extending MaGMA to support crashes and dynamic membership changes of MGMs is a natural extension to this kind of architecture. As explained above, this extension is not trivial and incorporates additional control overhead and more control mechanisms.

Another interesting direction for future work would be supporting hybrid networks, which are composed of both ad-hoc networks and access-point based networks [7]. In such networks, some MNs may be unable to directly communicate with any MGM. Therefore, one would need to delegate parts of the MGM's functionality to one of the ad-hoc nodes. This node would function both as the group manager representing the group members located in the ad-hoc network and as a relay forwarding the group's session packets to the group members it represents.

A third interesting future extension would be providing advanced services for group applications, e.g., floor control services, as specified in [38]. Since MaGMA is a distributed and dynamic architecture, supporting floor control in this kind of an environment can be more difficult than in traditional centralized groupware servers. Moreover, providing cross-platform session management for converged cellular/wireless networks can be challenging when nodes on the different networks have different characteristics, e.g., different compression standards, different delays, bandwidths, and jitter.

Chapter 10

Conclusions

We have presented MaGMA, an architecture for supporting real-time group services in integrated WiFi/cellular networks. MaGMA is the first comprehensive solution for groupware with seamless mobility and QoS support. MaGMA is very flexible and can co-exist with current as well as emerging wireless network technologies, it can support vertical handoff between WiFi/WiMAX and cellular modes of operation, and it is suitable for incremental deployment.

We have presented MaGMA's control and data planes, and have demonstrated MaGMA's efficiency using simulations and mathematical analysis. We have presented two solution types: a subscription model, suitable for lightweight servers and small groups, and a multicast overlay solution, which is more scalable and better supports large and dense groups. Finally, we described a proof-of-concept prototype implementation. We believe that MaGMA or similar services will play an important role in 3G and B3G environments.

References

[1] 3GPP. http://www.3gpp.org.

[2] 3GPP2. http://www.3gpp2.org.

[3] T. Anker, D. Breitgand, D. Dolev, and Z. Levy. CONGRESS: connection-oriented group address resolution services. In *Proceedings of SPIE on Broadband Networking Technologies*, pages 89–100. SPIE Press, 1997.

[4] A. Ballardie. Core Based Trees (CBT) Multicast Routing Architecture. RFC 2201 (Experimental), Sept. 1997.

[5] T. Ballardie, P. Francis, and J. Crowcroft. Core based trees (cbt). In *SIGCOMM '93: Conference proceedings on Communications architectures, protocols and applications*, pages 85–95. ACM Press, 1993.

[6] A. Bartoli. Group-based multicast and dynamic membership in wireless networks with incomplete spatial coverage. *Mobile Networks and Applications*, 3(2):175–188, 1998.

[7] E. M. Belding-Royer, Y. Sun, and C. E. Perkins. Global Connectivity for IPv4 Mobile Ad hoc Networks. Internet draft, IETF, draft-royer-manet-globalv4-00.txt, Nov. 2001.

[8] Y. Chawathe. Scattercast: an adaptable broadcast distribution framework. *Multimedia Syst.*, 9(1):104–118, 2003.

[9] V. Chikarmane, C. L. Williamson, R. B. Bunt, and W. L. Mackrell. Multicast support for mobile hosts using mobile IP: Design issues and proposed architecture. *Mob. Netw. Appl.*, 3(4):365–379, 1999.

[10] S. Deering and R. Hinden. Internet Protocol, Version 6 (IPv6) Specification. RFC 2460 (Draft Standard), Dec. 1998.

[11] R. Droms, J. Bound, B. Volz, T. Lemon, C. Perkins, and M. Carney. Dynamic Host Configuration Protocol for IPv6 (DHCPv6). RFC 3315 (Proposed Standard), July 2003.

[12] Flarion. 11 key design requirements for a mobile broadband network. Whitepaper, Feb. 2003.

[13] H. Gossain, C. Cordeiro, and D. Agrawal. Multicast: Wired to wireless. *Communications Magazine*, 40(6):116–123, June 2002.

[14] J. F. Huber. Mobile Next Generation Networks. *IEEE Multimedia*, 11(1):72–83, 2004.

[15] Jabber. http://www.jabber.com.

[16] D. Johnson, C. Perkins, and J. Arkko. Mobility Support in IPv6. RFC 3775 (Proposed Standard), June 2004.

[17] Microsoft. Microsoft Real Time Communication (RTC) Client. msdn.microsoft.com/downloads/list/clientapi.asp.

[18] S. mobile. Push to Talk over Cellular White Paper, 2004.

[19] G. Montenegro. Reverse Tunneling for Mobile IP, revised. RFC 3024 (Proposed Standard), Jan. 2001.

[20] J. Moy. Multicast Extensions to OSPF. RFC 1584 (Proposed Standard), Mar. 1994.

[21] J. Mysore and V. Bharghavan. A new multicasting-based architecture for internet host mobility. In *Proceedings of the 3rd annual ACM/IEEE international conference on Mobile computing and networking*, pages 161–172. ACM Press, 1997.

[22] National Institute of Standards and Technology. NIST SIP. http://snad.ncsl.nist.gov/proj/iptel/.

[23] Nextel. http://www.nextel.com/services/directconnect.shtml.

[24] Nokia. Push to Talk over Cellular – Real-time always-on voice service. http://www.nokia.com/poc/PoC_WP_A4_net.pdf.

[25] ns2 homepage. http://www.isi.edu/nsnam/ns.

[26] Open Mobile Alliance. http://www.openmobilealliance.org.

[27] C. Perkins. IP Mobility Support for IPv4. RFC 3344 (Proposed Standard), Aug. 2002.

[28] C. E. Perkins and D. B. Johnson. Mobility support in ipv6. In *MobiCom '96: Proceedings of the 2nd annual international conference on Mobile computing and networking*, pages 27–37. ACM Press, 1996.

[29] C. E. Perkins and D. B. Johnson. Route Optimization in Mobile IP. Internet draft, IETF, draft-ietf-mobileip-optim-11.txt, Sep. 2001.

[30] R. Prakash and R. Baldoni. Architecture for Group Communication in Mobile Systems. In *Proceedings of the The 17th IEEE Symposium on Reliable Distributed Systems*, pages 235–242. IEEE Computer Society, 1998.

[31] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. SIP: Session Initiation Protocol. RFC 3261 (Proposed Standard), June 2002. Updated by RFCs 3265, 3853.

[32] A. C. Snoeren and H. Balakrishnan. An end-to-end approach to host mobility. In *Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 155–166. ACM Press, 2000.

[33] L. Subramanian, I. Stoica, H. Balakrishnan, and R. H. Katz. OverQoS: offering internet QoS using overlays. *SIGCOMM Comput. Commun. Rev.*, 33(1):11–16, 2003.

[34] C. Tan and S. Pink. Mobicast: a multicast scheme for wireless networks. *Mob. Netw. Appl.*, 5(4):259–271, 2000.

[35] The Yankee Group. NG Push-to-Connect Provides Simple UI and Enriches User Experience, Sep. 2003.

[36] S. Thomson and T. Narten. IPv6 Stateless Address Autoconfiguration. RFC 2462 (Draft Standard), Dec. 1998.

[37] Togabi. http://www.togabi.com.

[38] I. T. Union. *Generic conference control. Recommendation T.124*. Standardization Sector of ITU, Geneva, Switzerland, Febuary 1998.

[39] Verizon Wireless. http://news.vzw.com/news/2003/08/pr2003-08-14.htm.

[40] D. Waitzman, C. Partridge, and S. Deering. Distance Vector Multicast Routing Protocol. RFC 1075 (Experimental), Nov. 1988.

[41] E. Wedlund and H. Schulzrinne. Mobility support using SIP. In *Proceedings of the 2nd ACM international workshop on Wireless mobile multimedia*, pages 76–82. ACM Press, 1999.

[42] WinterGreen. Push To Talk (PTT) Market Opportunities, Market Forecasts, and Market Strategies 2003-2008, Dec. 2003.

Appendix A

Analysis of MGMLeader

We now analyze the average control traffic overhead in the MGMleader protocol. We use the following notations:

- N - number of MNs in the network.
- D - number of domains and MGMs.
- g - number of groups.

We assume that MNs are uniformly distributed among the domains, and each MN uniformly chooses one of the available g groups to participate in. When an MN moves from one domain to another, there are two possible cases:

(i) The group is already represented in the new domain. In this case, the new MGM sends the move message to a group of MGMs representing the MN's group.

(ii) The group is absent from the new domain. In this case, the new MGM sends the move message to the former MGM. If the former MGM is not the group's coordinator, it forwards the message to the coordinator. The coordinator (whether it is the former MGM or not) then forwards the message to the group's MGMs and sends the view to the new MGM.

We denote the probability that i MNs out of n are in a specific domain

$$P_{MN}(n,i) = \binom{n}{i}\left(\frac{1}{D}\right)^{i}\left(\frac{D-1}{D}\right)^{n-i} \tag{A.1}$$

We note that $N-1$ is the maximum number of MNs in the new domain prior to the MN's arrival. We calculate the probability for each of the cases above:

(a) At least one of the local MNs chose this group, hence

$$p_a = \sum_{i=1}^{N-1} P_{MN}(N-1,i)\left(1-\left(1-\frac{1}{g}\right)^i\right) \tag{A.2}$$

(b) MNs from the new domain do not participate in the moving MN's group, therefore

$$p_b = \sum_{i=1}^{N-1} P_{MN}(N-1,i)\left(1-\frac{1}{g}\right)^i \tag{A.3}$$

In each of the aforementioned cases, the *move* message is sent to a subset of the MGMs, which have members in the moving MN's group. We calculate the probability that the moving MN's group is represented in a domain.

(a)

$$\begin{gathered} p_{a,N} = \frac{1}{p_a}\sum_{i=1}^{N-1} P_{MN}(N-1,i)\left(1-\left(1-\frac{1}{g}\right)^i\right) \\ \left[\sum_{j=1}^{N-i}\binom{N-i}{j}\left(\frac{1}{D-1}\right)^j\left(\frac{D-2}{D-1}\right)^{N-i-j}\left(1-\left(1-\frac{1}{g}\right)^j\right)\right] \end{gathered} \tag{A.4}$$

(b)

$$\begin{gathered} p_{b,N} = \frac{1}{p_b}\sum_{i=0}^{N-1} P_{MN}(N-1,i)\left(1-\frac{1}{g}\right)^i \\ \left[\sum_{j=1}^{N-i}\binom{N-i}{j}\left(\frac{1}{D-1}\right)^j\left(\frac{D-2}{D-1}\right)^{N-i-j}\left(1-\left(1-\frac{1}{g}\right)^j\right)\right] \end{gathered} \tag{A.5}$$

Using $p_{a,N}$,$p_{b,N}$, we calculate the average number of domains with members of the given group in each case. We note that the contribution of each domain to the average in cases (i),(ii) is $p_{a,N}$,$p_{b,N}$ respectively, thus the averages received

(i)

$$\overline{d}_a = (D-1)p_{a,N} \tag{A.6}$$

(ii)

$$\overline{d}_b = (D-1)p_{b,N} \tag{A.7}$$

We now evaluate the average number of messages sent in each case:

(i) The MGM sends messages to the appropriate MGMs (excluding itself), thus the average number of messages sent is

$$\overline{d}_a - 1 \tag{A.8}$$

(ii) The new MGM sends a message to the former MGM. If the former MGM is the coordinator, it send the message to group's MGMs (discarding a message to itself) and sends back a view message, for a total of

$$\overline{d}_b + 1 \tag{A.9}$$

If the former MGM is not the coordinator, it forwards the message to the group's coordinator, which acts as described above, for a total of

$$\overline{d}_b + 2 \tag{A.10}$$

An additional message is sent from the MN to the new MGM. We therefore get the following totals:

- The average number of messages sent in the case where the former MGM is also the coordinator
$$p_a\overline{d}_a + p_b\left(\overline{d}_b + 1\right) + 1 \tag{A.11}$$
- The average number of messages sent in the case where the former MGM is not the coordinator
$$p_a\overline{d}_a + p_b\left(\overline{d}_b + 2\right) + 1 \tag{A.12}$$

Appendix B

Analysis of MaGMA's Data Plane

We now analyze the average number of incoming streams per domain in the subscription and multicast overlay models. We use the same notations and assumptions of Appendix A. In the subscription model, the total number of incoming streams is the average number of MNs in a domain. As explained above, the probability that an MN chooses a certain domain is $\frac{1}{D}$. Calculating the average number of streams in domain:

$$\overline{S}_{subsc} = \sum_{i=0}^{N} \binom{N}{i} \left(\frac{1}{D}\right)^{i} \left(1 - \frac{1}{D}\right)^{N-i} = \frac{N}{D} \tag{B.1}$$

In the multicast overlay model, there is a single incoming stream for every group. Thus, the average number of incoming streams is equal to the average number of groups in a domain. We denote the probability that an MN chooses a certain group as p_{group}, we note that

$$p_{group} = \frac{1}{g} \tag{B.2}$$

Therefore the probability that an MN does not choose a specific group is $1 - p_{group}$. The probability that a certain group is represented in a domain with k MNs is

$$p_{dmn,k} = 1 - (1 - p_{group})^{k} \tag{B.3}$$

Thus, the probability that a group is represented in a domain is

$$p_{dmn} = \sum_{i=1}^{N} \left[\binom{N}{i} \left(\frac{1}{D}\right)^{i} \left(1 - \frac{1}{D}\right)^{N-i} p_{dmn,i} \right] \tag{B.4}$$

Each group contributes p_{dmn} to the average. Therefore, the average number of groups per domain is

$$\overline{S}_{overlay} = g \cdot p_{dmn} \tag{B.5}$$

Appendix C

MaGMA Multicast Overlay Correctness Proof

In this section we say that an *MGM joins the group* to denote the event that the MGM receives a *move* or *join* message of a group from a local MN, when it is not participating in the group. Likewise, we say that the *MGM leaves the group* if the MGM receives a *leave* from the last MN in this group in its domain.

We note that a *handoff* between MGMs can be divided into two actions: a join of the MGM that the MN has moved into its domain and a leave of the MGM that the MN has moved out of its domian. According to the current state of the MGMs, the handoff can be composed of one or both of the actions.

We note that as described in Section 5.7, all consecutive messages (*join* and *leave*) are received in the correct order.

We denote $\mathcal{S}$ as the set of MGMs in the network.

The proof's flow is depicted in Figure C.1.

Definition 1 (Live coordinator) *A live-coordinator, S_i, is an MGM that holds* coord$= i$. *A group with at least one live coordinator is a live group.*

Definition 2 (Sets in Group) *In group* g *at time* t*, there are 3 sets of MGMs:*

1. *$CoordView(t)$- MGMs that some coordinator processed their* join *message and no subsequent* leave *message before t.*

2. *$New(t)$- MGMs that are not in $CoordView(t)$ and sent* join *message at $t' < t$ with no subsequent* leave *message.*

3. *$Idle(t)$- the rest of the MGMs.*

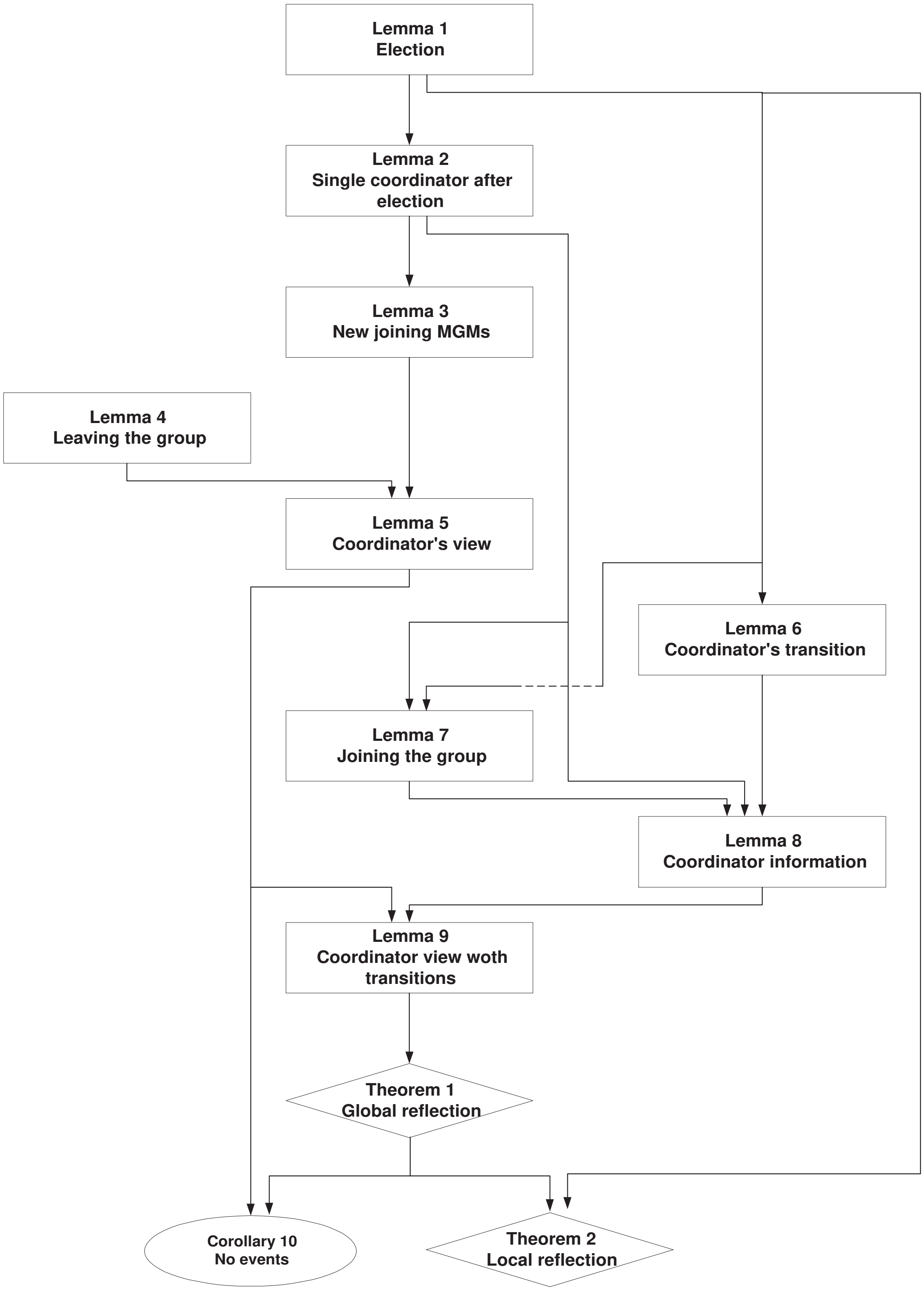


Figure C.1: The correctness proof flow.

Definition 3 (States in Group's Life) *We define three possible states of a group:*

1. *A group is in idle state at time t if $CoordView(t) = New(t) = \emptyset$.*

2. *A group is in election state at time t if $CoordView(t) = \emptyset$ and $New(t) \neq \emptyset$.*

3. *A group is in live state at time t if $CoordView(t) \neq \emptyset$.*

We define an active state as: active = election $\vee$ live. We define a period, $[t_1, t_2]$, as a time interval during which, $\forall t \in [t_1, t_2]$, the group does not change its state. A maximal period, $[t_1, t_2]$, is a period where at t_1 the group entered a new state and at t_2 changed its state again. Figure C.2 depicts a group's state transitions.

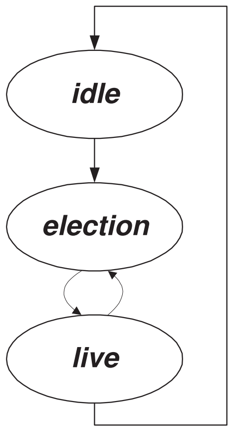


Figure C.2: A group's state transitions.

Definition 4 (View Assignment) *We define a view-assignment event as the settings of the group's view in Figure 5.3 line 8 or in Figure 5.5 line 3.*

Definition 5 (Active MGM) *We define an active MGM at time t, as an MGM that sent a* join *or* handoff *message before t and did not send a subsequent* leave *message before t, $MGM \in CoordView(t) \cup New(t)$.*

Lemma 1 (Election) *Let t_0 be a time where S_i sends a* join *message and until t_0 the group is idle. Then at $t_0 + 4\Delta$ S_i holds coord $\neq \perp$ and it assigns the group's view. Moreover, during $(t_0, t_0 + 5\Delta)$ no more than one MGM functions as the coordinator, and at $t_0 + 5\Delta$ there is a single MGM that is the live-coordinator, and all MGMs that participated in the election know its identity.*

Proof : The group enters an election period at t_0. Let $\mathcal{S}_e$ be the group of MGMs that broadcast *join* messages before receiving S_i's *join* message from t_0 onward ($S_i \in \mathcal{S}_e$). MGMs not in $\mathcal{S}_e$ hold *abstinence*= *true* from the reception of the first *join* message and 4Δ time units after the reception of the last *join* message, therefore they do not participate in the election procedure. All MGMs that are in $\mathcal{S}_e$ send *join* messages before $t_0 + \Delta$ and their timers expire no sooner than $t_0 + 4\Delta$. Thus, when their election timers expire, their *joiners-list*= $\mathcal{S}_e$. Therefore, since we use a deterministic function *coordSelect* to choose the coordinator (see Figure 5.3), exactly one of them becomes the coordinator of the group by $t_0 + \Delta + 4\Delta$. According to Figure 5.3 Line 10, the coordinator functions at least Δ time units, therefore if the coordinator sets its *coord* value at $t_0 + 4\Delta$, it still functions as the single live coordinator at $t_0 + 5\Delta$. In addition, all MGMs in $\mathcal{S}_e$ assign $\mathcal{S}_e$ to be their group's view, as described in Figure 5.3 line 8. When the elected coordinator assigns the view, the group enters a live period. □

Following Lemma 1 we note that:

(i) 5Δ time units after the first MN sends a *join* message there is a coordinator to the group.

(ii) Δ time units after a live period begins all MGMs that participated in the election hold *coord*$\neq\perp$.

In addition, if a live period begins at t_1 and let $t_1 < t < t_1 + \Delta$ be the first time where all MGMs that participated in the election (MGMs in *coordView(t_1)*) hold *coord*$\neq\perp$. Then following Lemma 1, (ii) and the 4Δ guard-time, MGMs not in *coordView(t_1)*, do not send *join* message before t.

Definition 6 (Coordinator transition) *We define coordinator transition as the procedure where the coordinator sends a* transition-request *to an MGM and receives either* transition-granted*, if the transition is succesful, or* transition-denied*, if the transition failed.*

Lemma 2 (Single coordinator after election) *Let $[t_1, t_2]$ be a maximal live period with no coordinator transitions. Then a single MGM, S_c, during this period functions as the live coordinator.*

Proof : We prove by induction on events.
(Base case) Lemma 1 - when a coordinator is elected it is single.
(Induction step) From the pseudo-code we know that MGMs that hold *coord*$\neq\perp$ do not change the *coord* value if there are no coordinator transitions. We examine the possible events and show that these events do not efect the induction's assumption.

(i) Let S_i be a new joining MGM that broadcasts a *join* message at time $t > t_1$. We know that S_i does not send a *join* message during the election, and when S_i sends its message all the MGMs that participated in the election and did not leave the group, hold the coordinator information. S_c receives the *join* message directly, and according to Figure 5.4 it sends back to S_i the group's view. By 2Δ time units after broadcasting the *join* message, S_i receives the group's view and coordinator information and sets *coord*$\leftarrow S_c$. Obviously, after receiving the *view* message from S_c, S_i does not change the *coord* value if there are no transitions. Therefore, only a single MGM function as the live coordinator.

(ii) Let S_i be an MGM leaving the group at $t > t_1$. Obviously, S_i is not the group's coordinator, therefore according to Figure 5.7 the participating MGMs do not change their *coord*.

(iii) Let S_i be a new joining MGM, that sends a *handoff* message to an MN's previous MGM, S_j, at $t > t_1$. From the assumptions we know that $\tau < 2\Delta$, therefore S_j does not leave the group before receiving S_i's *handoff* message. In addition as MN stays at least 10Δ time units in a domain, S_j holds the coordinator information when it receives the *handoff* message, by $t + \Delta$. Therefore it can forward the message to the coordinator. When S_c receives the *handoff* message, it sends the group's view to S_i. When S_i receives S_c's *view* message, it assigns the view and updates the *coord* to hold S_c, as described in Figure 5.5. In this case, by 3Δ time units after sending the *handoff* message S_i receives the group's view and coordinator information. Obviously, after receiving the *view* message from S_c, S_i does not change the *coord* value if there are no transitions. Therefore, only a single MGM function as the live coordinator.

□

Lemma 3 (New Joining MGMs) *Let $[t_1, t_2]$ be a maximal live period with no coordinator transitions, and let S_c be the group's coordinator. Then a new joining MGMs sets* coord$\leftarrow S_c$ *by 3Δ time units after sending* join *or* handoff *message.*

Proof : Based on Lemma 2, S_c is the single live coordinator during the live period. Let $t > t_1$ be the time where S_i sends the *join* or the *handoff* message. Then S_i receives the group's view and coordinator information by 2Δ time units in case of a join scenario and by 3Δ time units in case of a move scenario. □

Definition 7 (Current membership status reflection) *View v correctly reflects S_i's current membership status if $S_i \in v$ iff S_i is active.*

Lemma 4 (Leaving the group) *An MGM does not send a* leave *message before its membership status is correctly reflected in the coordinator's* groupView*.*

Proof : This follows from Figure 5.6 Line 2, an MGM does not send a *leave* message before receiving the group's view from the coordinator. □

Lemma 5 (Coordinator's view) *Let $[t_1, t_2]$ be a live period with no coordinator transitions, where at t_1 the group enters the live period. Then every MGM, that its last event occured at $t \in (t_1, t_2 - 4\Delta)$ is correctly reflected in the coordinator's group view from $t + 4\Delta$ until t_2.*

Proof : Based on Lemma 1, after the election ends and a live period begins at t_1, there is a single coordinator and the coordinator knows the identity of all the MGMs that participated in the election. According to Lemma 2, if there are no transitions the coordinator's identity does not change, and till there is a transition there is a single live coordinator. We note that the coordinator's view and status changes only as a result of join, leave and move events. We denote the group's coordinator as S_c.

We now examine the actions taken according to the type of event:

(i) Let S_i be an MGM that broadcasts a *join* message at time $t \in (t_1, t_2 - 4\Delta)$. The message is sent to $\mathcal{S}$, and by $t+\Delta$ S_c receives the message and adds S_i to its *groupView*.

(ii) Let S_i be an MGM that sends a *leave* message at time $t \in (t_1, t_2 - 4\Delta)$ to MGMs in its *groupView*. According to Lemma 4, before leaving the group S_i's status is correctly represented in the coordinator's group view. Due to the fact that there are no coordinator transitions, it is clear that $S_c \in$*groupView*, thus S_c receives the message by $t + \Delta$ and removes S_i from its *groupView*.

(iii) Let MN_i be an MN moving from S_j to S_i at time $t \in (t_1, t_2 - 4\Delta)$. MN_i sends a *move* message to S_i, S_i receives the message by time $t + \Delta$. If S_i needs to join the group it sends a *handoff* message to S_j, which receives the message by $t + 2\Delta$. From the assumptions we know that $\tau < 2\Delta$, therefore S_j does not leave the group before receiving S_i's *handoff* message. In addition as MN stays at least 10Δ time units in a domain, S_j holds the coordinator information when it receives the *handoff* message. Therefore it can forward the message to the coordinator. S_j forwards the *handoff* message to S_c. S_c receives the message by $t + 3\Delta$ and adds S_i to its *groupView*. In addition, if S_j needs to leave the group, it sends a *leave* message to all MGMs in its *groupView* (including S_c). Therefore, according to (ii) S_c removes S_j from its *groupView* by $t + 3\Delta$. If S_i does not need to join the group, S_j detects the disconnection of MN_i by $t + \tau$. If it then needs to leave the group it sends a *leave* message, as described above, that is received by S_c by $t+\tau+\Delta$. Since $2\Delta < \tau < 3\Delta$, the lemma follows.

Due to the fact that this is the last event of S_i until t_2, and it does not change its participation status, until t_2 S_i's membership status is correctly reflected in the coordinator's group view. □

Lemma 6 (Coordinator's transition) *Let* $[t_1, t_2]$ *be a maximal live period of a group, and let* S_c *be the group's live coordinator initiating a transition at time* $t \in (t_1, t_2 - 2\Delta)$*. When the transition procedure ends (successfully or not), there is a single MGM that functions as the live coordinator. In addition, from* $t+2\Delta$ *all MGMs in* $\mathrm{CoordView}(t)$ *hold that coordinator's identity.*

Proof : We prove by induction on transitions.
(Base case) Based on Lemma 1, after the election there is a single live coordinator. We now examine the first transition after the election, let S_n be the MGM chosen by S_c to be the next coordinator. We first examine the case where the transition succeeds. S_c sends a *transition-request* to S_n at time t. S_n receives this message by $t + \Delta$, sends a *transition-granted* message to S_c, and sends to MGMs in $CoordView(t)$ *new-coord* messages. These MGMs receive the messages by $t+2\Delta$, remove S_c from their *groupView*, and set their *coord* to hold S_n, as described in Figure 5.8. Upon receiving *transition-granted*, S_c sets *coord*$\leftarrow\perp$. Due to the fact that the group's coordinator must remain in the group at least Δ time units, after S_c's *coord* initialization there is a single MGM that function as the group's coordinator.

We now examine the case where the transition failes, this occurs when S_n replies with a *transition-denied*. Therefore, when S_c receives the message it is the only MGM that functions as the group's coordinator.

Note that, if $\|CoordView(t)\| = 1$, meaning only the coordinator participates in the group, the coordinator's $\|localView\| = 1$ and the coordinator receives, at time t, a *leave* message from the MN. Then it leaves the group without sending any message. It is clear that $t = t_2$ or t_2 is Δ time units after the last transition, and that the coordinator is the last MGM to leave the group.

(Induction step) By the Base case, when the first transition ends at time t, there is a single live coordinator. In addition, by the time where the new coordinator is allowed to leave the group, $t' > t$, all MGMs in *coordView(t')* hold or will hold (in case of join events) the new coordinator identity. Therefore, according to the status of the participating MGMs and the coordinator's identity, the next transition's behaviour is similar to the previous one. □

Lemma 7 (Joining the group) *Let* $[t_1, t_2]$ *be a maximal live period, let* S_i *be a new joining MGM either broadcasting a* join *message or sending a* handoff *message to another MGM at time* $t \in (t_1, t_2)$*, and* S_i *does not send a* leave *message before* $t + 6\Delta$*. Then by* $t + 6\Delta$*,* S_i *sets the group's view and coordinator information from a live coordinator, and* $S_i \in \mathrm{coordView}(t + 6\Delta)$*, unless it leaves the group.*

Proof : By Lemma 2, while there are no transitions, the joining MGM receives the group's view and the coordinator identity by $t + 3\Delta$.

Let S_c be the group's coordinator initiating a transition procedure at t' that ends at $t'' > t'$, and let S_n be the next coordinator selected by S_c. We first examine the scenario where S_i joins the group by broadcasting a *join* message. In this scenario, S_c receives the *join* message by $t + \Delta$, $t' < t + \Delta < t''$. S_c buffers and forwards the message to S_n. S_n receives the message by $t + 2\Delta$. If it grants the transition, it adds S_i to its *groupView* and sends to S_i the group's view. Thus, by $t + 3\Delta$, S_i receives the group's view and coordinator identity from S_n.

If S_c receives the *join* message during $(t'', t''+\Delta)$ interval, after S_n granted the transition. S_c tunnels the message towards S_n. S_n receives the message by $t + 2\Delta$, S_n adds S_i to its *groupView*, and by $t + 3\Delta$, S_i receives the group's view and coordinator identity from S_n.

We note that due to the fact that S_i broadcasts the *join* message, S_n can receive the message directly from S_i after granting the transition. According to Figure 5.2, S_n processes the message only once and discards the tunneled message. Thus, by $t + \Delta$, S_n adds S_i to its *groupView*, and by $t + 2\Delta$, S_i receives the group's view and coordinator identity from S_n.

If S_n denies the transition, then by $t + 3\Delta$, S_c receives the *transition-denied* message and processes the buffered messages. Thus, by $t + 4\Delta$, S_i receives the group's view and the coordinator's identity from S_c.

We now examine the second scenario where S_i joins the group due to a movement scenario. S_i sends a *handoff* message at time t to the previous MGM of the MN, denoted as S_j. From the assumptions we know that $2\Delta < \tau < 3\Delta$, therefore S_j does not leave the group before receiving S_i's *handoff* message. In addition as MN stays at least 10Δ time units in a domain, S_j holds the coordinator information when it receives the *handoff* message, by $t + \Delta$. Therefore it can forward the message to the coordinator. S_j forwards to S_c the *handoff* message. S_c receives the message by $t + 2\Delta$ and then acts as described above in the join scenario, thus if the transition is successful, S_i receives the group's view and coordinator indentity by $t+5\Delta$ from S_n, and if the transition is denied, then by $t+6\Delta$, it receives the group's view and coordinator indentity from S_c.

Another possible scenario occurs when $t \in (t_2 - \Delta, t_2)$. If S_i sends a *join* message at t, it is possible that the coordinator does not receive the message by t_2, and a new election procedure begins. Therefore, according to Lemma 1 by $t+4\Delta$ S_i sets the group's view and coordinator information, and $S_i \in$ *coordView(*$t+5\Delta$*)*. □

Lemma 8 (Coordinator information) *Let* $[t_1, t_2]$ *be a live period, and let* $S_i \in \text{coordView}(t)$, $t \in (t_1, t_2)$. *Then from* $t + \Delta$, S_i*'s* coord *value holds the current coordinator identity or the identity of a previous coordinator that holds a tunnel towards the current coordinator.*

Proof : According to Lemma 2 in a live period with no transitions there is a single MGM that functions as the group's coordinator. Therefore, if the first transition occurs at time t'_1 and denoting the group's coordinator in this time interval as S_c, then for $t' \in (t_1, t'_1)$, for all $S_i \in$ *coordView(*t'*)*: *coord*$=S_c$.

We now examine the first transition that begins at t'_1. If the transition is denied, then when the transition ends, at $t'_1 < t''_1 < t'_1 + 2\Delta$, MGMs in *coordView(*$t''_1$*)* hold the identity of the current coordinator (S_c) and according to Lemma 7 new joining MGMs recieve the view message from S_c, because the new coordinator denied the transition and did not processed new *join* messages. If the transition succeeds, denoting the new coordinator as S_n, then by $t'_1 + \Delta$ S_n receives the *transition-request* message, assigns *coord*$\leftarrow S_n$ and the group view, sends *transition-granted* message to S_c and sends a *new-coord* message to *coordView(*t'_1*)*. S_c receives the *transition-granted* message by $t'_1 + 2\Delta$ and sets a tunnel towards S_n for Δ time units. MGMs in *coordView(*t'_1*)* receives the *new-coord* message by $t'_1 + 2\Delta$, during this time S_n does not initiate a transition procedure, thus these MGMs' *coord* values hold the identity of the previous coordinator (S_c) that holds a tunnel towards the new coordinator (S_n). By Lemma 6, by $t'_1 + 2\Delta$ all MGMs in *coordView(*t'_1*)* hold the identity of the new coordinator, MGMs that joins the group during a successful transition receives the identity of the new coordinator. Thus Δ time units after S_n replies with a *transition-granted* mesasge MGMs in *coordView* hold the identity of the new coordinator. Obviously, after that time S_n can initiate a new transition, at $t'_2 > t'_1 + \Delta$, therefore the proof is similar to the first transtion as described above and thus the lemma follows. □

Lemma 9 (Coordinator view with transition) *Let* $[t_1, t_2]$ *be a live period, where at* t_1 *the group enters the live period. Then every MGM, that its last event occurs at* $t \in (t_1, t_2)$, *is correctly reflected in the coordinator's group view from* $t + 6\Delta$ *and until* t_2.

Proof : We note that if the last event of an MGM occurs during $(t_1, t_2 - 4\Delta)$ and there are no transitions then based on Lemma 5 the MGM's membership status is correctly reflected in the coordinator' view from $t + 4\Delta$.

We now examine the first transition and the three possible events for $t \in (t_1, t_2 - 6\Delta)$. Based on Lemma 6, there is a single live coordinator that processes messages sent due to join, leave, and move events. By Lemma 8, at time t MGMs in *coordView(t − Δ)* hold, either the coordinator identity or the previous coordinator identity that has established a tunnel towards the new coordinator. We now examine the three events that can affect the coordinator's view:

(i) By Lemma 7, join events are reflected in the coordinator's view within 6Δ time units.

(ii) Let S_i be an MGM sending a *leave* message at time $t \in (t_1, t_2)$. According to Lemma 8, S_i's *coord* holds either the current coordinator's identity or the previous coordinator's identity that has a tunnel towards the current coordinator. Thus, in the first case, the current coordinator, denoted as S_c, receives the *leave* message by $t + \Delta$ and removes S_i from its *groupView*. In the latter case, S_c forwards the message to the new coordinator, denoted as S_n. If S_n grants the transition then by $t + 2\Delta$ it processes the *leave* message. If it denies the transition, then it sends a *transition-denied* to S_c, upon receiving the message by $t + 3\Delta$, S_c processes the S_i's buffered *leave* message. Thus, by 3Δ the leave event is reflected in the coordinator's view.

(iii) Let MN_i be an MN moving from S_j to S_i at time $t \in (t_1, t_2)$. MN_i sends a *move* message to S_i, S_i receives the message by $t+\Delta$. If S_i needs to join the group it sends a *handoff* message to S_j. S_j receives the message by $t+2\Delta$ and forwards the message to S_c. According to Lemma 8, S_j's *coord* holds either the current coordinator's identity, denoted as S_c, or the previous coordinator identity that holds a tunnel towards the current coordinator. In the first case, S_c receives the message by $t+3\Delta$ and adds S_i to its *groupView*. In the latter case, S_c receives the message by $t + 3\Delta$ and forwards the message to the new coordinator, denoted as S_n. S_n receives the message by $t + 4\Delta$. If it grants the transition then it adds S_i to its *groupView*. On the other hand, if it denies the transition, S_c processes S_i's buffered message by $t + 5\Delta$, as described in (i). In addition if S_j needs to leave the group it sends a *leave* message to all MGMs in its *groupView*. As described in (ii), the event is reflected in the coordinator's view by $t+5\Delta$. If S_i does not need to join the group, S_j detects the disconnection of MN_i by $t + \tau$. If it then needs to leave the group it sends a *leave* message, as described in (ii), this event is reflected in the coordinator's view by $t + \tau + 3\Delta$. As $2\Delta < \tau < 3\Delta$, the lemma follows.

Based on Lemma 8, we know that Δ time units after the first transition, denoted as t' all MGMs in *coordView(t')* hold the current live coordinator. In addition, the new coordinator can initiate a new transition only Δ time units after granting the transition. Thus, based on the transition description above, the lemma follows for the following transitions.

□

Theorem 1 (Global reflection) *Let $[t_1, t_2]$ be a live period of a group, where at t_1 the group enters the live period. Then every MGM that its last event occurs at $t \in (t_1, t_2)$, is correctly reflected from $t + 7\Delta$ until t_2 in the* groupView *of MGMs in* coordView(t')*, for all $t' \geq t + 7\Delta$.*

Proof : We first examine the scenarios where $t \in (t_1, t_2 - 7\Delta)$. According to Lemma 9, every MGM, that has no events from time t, is correctly reflected in the coordinator's *groupView* by 6Δ. We note that MGMs, that participate in the group, receive messages directly from the sending MGMs or through a tunnel from the group's coordinator. We examine the three events (join, leave, and move) that can modify an MGM's *groupView*:

(i) When an MGM joins the group it broadcasts a *join* message. If S_i broadcasts a *join* message at time $t \in (t_1, t_2)$ MGMs receive the message by $t + \Delta$. Thus, MGMs in *coordView(t)* and stay active add S_i to their *groupView* by Δ time units. In addition, the group's coordinator receives the message, also by $t + \Delta$. By Lemma 9, every MGM's membership status is reflected in the coordinator's *groupView* by 6Δ time units. MGMs that are in *active-tunnels* receive the message through a tunnel from the coordinator. Thus, if the coordinator processes the message at time t', the by $t' + \Delta$, all MGMs in *coordView(t')* processes the message. From t' onward, all new joining MGMs receive the group view that include S_i.

(ii) Let S_i be an MGM that leaves the group at time t. S_i sends a *leave* message to MGMs in its *groupView*. These MGMs receive the message by $t + \Delta$. In addition, MGMs in *active-tunnels* receive the message through a tunnel from the group's coordinator. Thus, by Lemma 9, if the coordinator processes the message by $t + 6\Delta$, then by $t + 7\Delta$ all MGMs in *coordView(t')* processes the message.

(iii) We note that every move event can be divided into a join and leave event. Thus, according to (i) and (ii), if the coordinator processes the message at $t + 6\Delta$ then by $t + 7\Delta$ all MGMs in *coordView(t')* processes *join* and/or *leave* messages.

□

Corollary 10 (No events) *If there are no events during a maximal live period* $[t_1, \infty)$*, then the membership status of the MGMs in* $\mathcal{S}$ *are correctly represented in the views of MGMs in* coordView(t)*,* $\forall t > t_1 + \Delta$*.*

Proof : This is immediate from Lemma 5. □

Theorem 2 (Local reflection) *If an MN joins a group at time* t*, then from* $t + 10\Delta$ *its MGM membership status is correctly reflected in the views of MGMs in* coordView(t')*, for all* $t' > t + 10\Delta$*.*

Proof : We examine two scenarios: one when the MN joins before or during an ongoing election and the other when the MN joins during a live period.

If the MN joins during a live period and its MGM does not participate in the group then by the assumption on message propogation and by Theorem 1 by 8Δ the MGM's status is correctly reflected. If the MN's MGM already participates in the group, i.e. it already sent a *join* message, then obviously the theorem follows.

If the MGM receives the MN's *join* message and its *abstinence*=false and currently the group is in idle state, then it initiates an election. By Lemma 1 and by Theorem 1 the theorem follows. If the MGM receives the MN's *join* message and its *abstinence*=true, meaning there is an ongoing election, then it after 5Δ time units the MGM sends the *join* message, and by Theorem 1 this theorem follows. □

תמיכה בתקשורת קבוצתית ברשתות עם משתמשים ניידים

נדב לביא

תמיכה בתקשורת קבוצתית ברשתות עם משתמשים ניידים

חיבור על מחקר

לשם מילוי חלקי של הדרישות לקבלת תואר
מגיסטר למדעים
הנדסת חשמל

נדב לביא

הוגש לסנט הטכניון מכון טכנולוגי לישראל

סיון תשס"ה   חיפה   יוני 2005

המחקר נעשה בהדרכת פרופ' ישראל צידון וד"ר עדית קידר
הפקולטה להנדסת חשמל

הכרת תודה

אני רוצה להודות למשפחתי על כך שהאמינו בי ועל אהבתם ותמיכתם.

אני רוצה להודות מקרב לב למנחים שלי, פרופ' ישראל צידון וד"ר עדית קידר, על ההקשבה והסבלנות, ועל כך שלא חסכו מזמנם וממרצם להצלחת המחקר.

אני מודה לפרופ' ישראל צידון על כך שהעניק מהידע ומנסיונו הרב באקדמיה ובתעשיה, וכל זאת בדרך ההומור האופיינית לו, ולימד אותי לחבר בין התאוריה למעשה.

אני מודה לד"ר עדית קידר על כך שדחפה אותי לשאוף לשלמות ולימדה אותי מחקר מדעי מהו, וכל זאת בדרכי נועם.

ברצוני להודות ליורם אור־חן, על כך שהיה בין המשפיעים עלי להמשיך את לימודי לתואר שני, ועל עצותיו המועילות.

אחרונים חביבים, אני מודה לחברי ללימודים, גל בדישי, ארן ברגמן וישכר (זיגי) ולטר, על ההפריה ההדדית, החברות והתמיכה, ועל כך שהפכו את השהיה בטכניון לנעימה וביתית.

כמו כן, אני מודה ל־Anne Abel and Benjamin Pinkel Fellowship על התמיכה הכספית הנדיבה בהשתלמותי.

אני מודה לטכניון על התמיכה הכספית הנדיבה בהשתלמותי

תוכן עניינים

רשימת איורים

רשימת טבלאות

תקציר

הזמינות של האינטרנט הובילה לשימוש רב באפליקציות לתקשורת־קבוצתית (chat, ICQ, NetMeeting, Exchange, Lotus Notes, Webex וכו'). עם התקדמות טכנולוגיית הרשתות האלחוטיות, השימוש באפליקציות הללו ברשתות הסלולר ובשאר הרשתות האלחוטיות הפך לפופולרי במיוחד [35]. מספר ספקיות סלולר גדולות כמו Nextel, Verizon, Orange כבר מספקות שירותי תקשורת־קבוצתית מסוג לחץ־ודבר (push-to-talk, PTT) [23, 39]. שוק ה־PTT, מנה בשנת 2003 2.3 מיליון משתמשים ושוויו היה כ־84 מיליון דולר, והוא צפוי לגדול לשוק של כ־340 מיליון משתמשים ולשווי של 10.1 מיליארד דולר עד לשנת 2008 [42]. בזמן ש־PTT מוגבל לשידור קול בלבד, רשתות ה־B3G (beyond-3G) העתידיות יאפשרו חוויה עשירה המשלבת גם היבטים של אפליקציות זמן־אמת בעולם התקשרות הקבוצתית.

ארכיטקטורת האינטרנט מתחילה לספק את התמיכה הנדרשת על ידי אפליקציות מסוג זמן־אמת, הדורשות איכות־שירות (QoS) בתחנות הקצה הנייחות. הדרישות הנ"ל גררו תכנון פרוטוקולים חדשים כגון: Diffserv, RSVP ו־MPLS לתמיכה באיכות־שירות על גבי האינטרנט וכן RTP, H.323, MGCP ו־SIP לתמיכה ישירה באפליקציות מסוג זמן־אמת.

רשתות האלחוט שהפכו לנפוצות במיוחד ברחבי העולם כולו, מאפשרות גישה אלחוטית לרשת האינטרנט. ישנם מספר תקנים מובילים כמו WiFi ו־WiMAX לתקשורת אלחוטית ומספר רב של קבוצות עבודה העוסקות באספקטים שונים בנושאים הקשורים לתמיכה במשתמשים אלחוטיים וניידים ברשתות אלו. כיום אין תמיכה מספקת במשתמשים ניידים ברשתות אלחוט מבוססות פרוטוקול ה־IP המעוניינים בתקשורת זמן־אמת קבוצתית. התקן המוצע על ידי ה־IETF הינו ה־Mobile IP (MIP) המשלב, בצורה אינהרנטית, תקורה גבוהה ואינו מספק איכות־שירות כפי הנדרש ברשתות כיום.

ברור כי השלב הבא הוא התמזגות כל רשתות התקשורת, רשת האינטרנט ורשתות האלחוט המסופחות לה יחד עם רשת התקשורת הסלולרית. ישנם צידוקים רבים לכך שרשתות האלחוט העתידיות יאמצו את ארכיטקטורת ה־TCP/IP ואת המנגנונים הנלווים לתמיכה בתקשורת זמן־אמת, איכות־שירות וניידות. כבר היום מאמצים עקרונות רבים מעולם ה־TCP/IP, לא רק עבור שכבת התעבורה (transport), אלא גם עבור השכבות הגבוהות, לדוגמה, השימוש הנרחב בפרוטוקול SIP [1, 2, 26]. המגמה הזאת מאיצה את שילוב הרשתות הסלולר עם רשת האינטרנט [14]. במקביל קיימות רשתות גישה אלחוטיות, זולות ובעלות קצבים גבוהים. אלה מאפשרות גישה אלחוטית לרשתות מבוססות IP, והופכות להיות זמינות

יותר ויותר דרך נקודות גישה מסוג Wi-Fi ו־WiMAX. מכשירי־הקצה חופשיים ממגבלות החיווט, הופכים להיות קטנים וקלים יותר, וכמובן השימוש בהם הופך לפשוט יותר. שתי המגמות המקבילות הללו, מובילות למיזוג הדרגתי בין רשתות הסלולר ורשתות ה־IP, שבעבר היו מנותקות לחלוטין. המיזוג הנ"ל מתבצע הן ברמת מכשירי־הקצה והן ברמת ארכיטקטורת הרשתות. כמובן שעקב חשיבותן של אפליקציות מסוג תקשורת קבוצתית, הרשת הממוזגת העתידית תהיה חייבת לתמוך בתקשורת מסוג זה הן עבור תקשורת זמן־אמת והן עבור תקשורת מידע.

ברור כי החוליה החסרה באבולציה של הרשתות האלחוטיות היא העדר תמיכה מקיפה בניהול קבוצות וכמובן העדר תמיכה באפליקציות זמן־אמת בתקן הניידות של ה־IP. האפליקציות העתידיות האלה תהיינה חייבות בבסיס מוצק לתמיכה במשתמשים הניידים.

עבודת המחקר מציגה את MaGMA (Mobility and Group Management Architecture) ־ ארכיטקטורה חדשנית לניהול קבוצות ברשתות אלחוטיות המקושרות דרך רשת האינטרנט. MaGMA תומכת: בניהול סקלאבילי של קבוצות, בתמיכה בניידות, בתמיכה ב־handoff ובאיכות־השירות, ומספקת פתרון כולל לעולם הרשתות האלחוטיות. פתרון מסוג זה יהיה חלק בלתי נפרד מארכיטקטורת רשתות ה־3G וה־B3G. לא ידוע לנו על פתרון קודם העונה לכל הצרכים שתוארו לעיל.

הארכיטקטורה של MaGMA מורכבת מאוסף של שרתים הנקראים MGMs (Mobile Group Managers) וממכשירי־קצה (Mobile Nodes, MNs). השרתים מנהלים את הקבוצה ויכולים לתמוך בהפצת מידע על־גבי רשת overlay. כל MN מקבל שירות מ־MGM הקרוב אליו. פיתוח פרוטוקול ניהול קבוצות יעיל , סקלאבילי ומבוזר המאפשר גם ניידות משתמשים הוא מאתגר בחינה טכנית.

המחקר בחן שתי גישות לניהול קבוצות תחת MaGMA: הגישה הראשונה נקראת subscription model, בה ה־MGMs תומכים אך ורק בניהול הקבוצות. המשתמשים יכולים לקבל את רשימת המשתתפים מהשרתים ובעזרתה לשלוח לכל משתתף הודעות בצורה ישירה ללא כל עזרה של השרתים. הגישה השניה נקראת multicast overlay model, בה השרתים לא רק מנהלים את הקבוצה אלא גם משמשים להעברת הודעות בשיטת ה־multicast על גבי ה־overlay שהשרתים יוצרים ביניהם.

כאמור, במודל ה־subscription השרתים (MGMs) אחראים אך ורק על ניהול הקבוצה ואינם משתתפים בהעברת מידע. כאשר משתמש מעוניין לשלוח חבילות מידע לשאר חברי הקבוצה הוא תחילה מבקש מהשרת המנהל את הדומיין (domain) בו הוא נמצא את רשימת חברי הקבוצה. כאשר הוא מקבל את הרשימה הוא שולח את חבילות המידע ישירות לחברי הקבוצה. מודל זה משלב את גישת peer-to-peer (P2P), הגישה הפופולארית כיום בעולם האינטרנט, ועל כן מחייבת שמירת רשימת חברי הקבוצה ברמת המשתמשים (MNs). מודל זה מתאים במיוחד לקבוצות קטנות, מאחר וישנה בעיה לתמוך במספר רב של sessions במכשיר־קצה יחיד, וכן מתאים המודל לארכיטקטורה עם שרתים פשוטים מאחר והשרתים אחראים רק על ניהול הקבוצה ולא משמשים להעברת מידע. מודל זה לא מתאים לקבוצות

גדולות ולמכשירי־קצה עם מגבלות סוללה. המחקר הראה כי בקבוצות גדולות שימוש במודל זה עלול להעמיס את הרשת מאחר ומספר ה־sessions הנכנסים לדומיין (domain) גדל בצורה לינארית ביחס למספר המשתמשים הנמצאים בדומיין.

מודל ה־multicast overlay משפר באופן ניכר את השירות על ידי השימוש בשרתים להעברת מידע. במודל זה, כל משתמש המעוניין לשלוח חבילת מידע לקבוצה שולח עותק יחיד של החבילה לשרת האחראי על הדומיין בו הוא נמצא. השרת מקדם את ההודעה לשאר השרתים שבדומיין שלהם נמצאים משתמשים החברים בקבוצה. השרתים מקדמים את ההודעה למשתמשים הנמצאים תחת אחריותם והחברים בקבוצה. במודל זה נדרש לנהל שתי רשימות לכל קבוצה ובכל שרת. הראשונה, רשימת המשתמשים הנמצאים בדומיין וגם חברים בקבוצה, והשניה, רשימת שאר השרתים שלהם משתמשים החברים בקבוצה. המחקר הראה כי מודל ה־multicast overlay מספק פתרון סקלאבילי וגמיש הן לניהול הקבוצות והן להעברת מידע.

למרות ששני המודלים (subscription ו־multicast overlay) משתמשים בשתי גישות שונות לניהול הקבוצה ולהעברת מידע, הם יכולים להיות בשימוש, בעת ובעונה אחת ברשת. ברור כי מספר קבוצות יכולות להתקיים ברשת בזמן נתון וכל קבוצה על פי גודלה ומאפייניה תשתמש במודל המתאים לה.

בעבודת מחקר זו אנו מציגים: תיאור כללי של מודל ה־subscription, תיאור פורמאלי ומפורט של מודל ה־multicast overlay הכולל פסאודו־קוד, וכן הוכחת נכונות לפרוטוקול ניהול הקבוצה. כמו כן המחקר מציג הערכת ביצועים על־ידי אנליזה מתמטית וסימולציות שנערכו בעזרת ns2 [25] עבור שתי הגישות. הסימולציות בחנו את ביצועי שכבת הבקרה וניהול הקבוצות ואת ביצועי שכבת התעבורה.

בשכבת הבקרה נבחנו ביצועי פרוטוקולי ניהול הקבוצות עבור כל אחד מהארועים האפשריים ברשת: הצטרפות, עזיבה, תזוזה והתנתקות של משתמש. נמדדו מספר הודעות הבקרה הנשלחות וזמן התייצבות רשימת חברי הקבוצה עקב אחד מארבעת הארועים.

כמו כן נבחנו ביצועי שכבת התעבורה של MaGMA לעומת ביצועי MIP, מבחינת זמן הגעה של חבילות (delay) ואיבודי חבילות. כמו כן נערכה השוואת גודל חבילה (header overhead) בין MaGMA לפתרונות אחרים כמו: MIP בדומיין הבית ומחוצה לו, IP multicast, MIP multicast ו־unicast. יתר על כן נערכה השוואה בין שני המודלים של MaGMA מבחינת session overhead. עבודת המחקר מציגה התכנות, עבור מודל ה־multicast overlay בעזרת אב־טיפוס מבוסס SIP ברשת Wi-Fi אלחוטית עם מכשירי־קצה מסוג iPAQ.

תוצאות המחקר מראות כי MaGMA עדיפה לאין שעור בהשוואה לפתרון MIP מבחינת ביצועי שכבת התעבורה. MaGMA אינה משתמשת בפתרון מסוג triangle routing כפי ש־MIP משתמש, ועל כן ההשהייה נמוכה ומספר איבודי החבילות עקב תזוזה קטן יותר. כן הראה המחקר כי מודל ה־multicast overlay סקלאבילי יותר ממודל ה־subscription הן בשכבת התעבורה והן בשכבת הבקרה וכי MaGMA הינה ארכיטקטורה המספקת פתרון מקיף לתמיכה בתקשורת קבוצתית ברשתות עם משתמשים ניידים. לסיכום נדגיש כי MaGMA

מאפשרת ניהול סקלאבילי של קבוצות וכן מאפשרת העברת תעבורה בין משתמשים ניידים המשנים את מיקומם ואת כתובתם ברשת תוך שמירה על איכות־שירות על מנת לתמוך באפליקציות זמן־אמת.