Integrated Mobility and Service Management for Network Cost Minimization in Wireless Mesh Networks

Yinan Li

Dissertation submitted to the Faculty of the Virginia Polytechnic Institute and State University in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Computer Science and Applications

Committee:
Ing-Ray Chen, Chair
Denis Gracanin
Chang-Tien Lu
Hazhir Rahmandad
Naren Ramakrishnan

April 30, 2012
Falls Church, Virginia

Keywords: wireless mesh networks, integrated mobility and service management, mobile data access, cache consistency management, mobile multicast services, security, performance analysis.

©Copyright 2012, Yinan Li
Abstract

Yinan Li

In this dissertation research, we design and analyze integrated mobility and service management for network cost minimization in Wireless Mesh Networks (WMNs). We first investigate the problem of mobility management in WMNs for which we propose two efficient per-user mobility management schemes based on pointer forwarding, and then a third one that integrates routing-based location update and pointer forwarding for further performance improvement.

We further study integrated mobility and service management for which we propose protocols that support efficient mobile data access services with cache consistency management, and mobile multicast services. We also investigate reliable and secure integrated mobility and service management in WMNs, and apply the idea to the design of a protocol for secure and reliable mobile multicast. The most salient feature of our protocols is that they are optimal on a per-user basis (or on a per-group basis for mobile multicast), that is, the overall network communication cost is minimized for each individual user (or group). Per-user based optimization is critical because mobile users normally have vastly different mobility and service characteristics. Thus, the overall cost saving due to per-user based optimization is cumulatively significant with an increasing mobile user population.

To evaluate the performance of our proposed protocols, we develop mathematical models and computational procedures used to compute the network communication cost incurred and build simulation systems for validating the results obtained from analytical modeling. We identify optimal design settings under which the network cost is minimized for our mobility and service management protocols in WMNs. Intensive comparative performance studies are carried out to compare our protocols with existing work in the literature. The results show that our protocols significantly outperform existing protocols under identical environmental and operational settings.

We extend the design notion of integrated mobility and service management for cost minimization to MANETs and propose a scalable dual-region mobility management scheme for location-based routing. The basic design is to use local regions to complement home regions and have mobile nodes in the home region of a mobile node serve as location servers for that node. We develop a mathematical model to derive the optimal home region and local region sizes under which overall network cost is minimized. Through a comparative performance study, we show that dual-region mobility management outperforms existing schemes based on static home regions.
Acknowledgments

I want to sincerely express my gratitude to all the people who have supported me during my Ph.D. study. Without their encouragement and help, it impossible for me to complete this dissertation.

First of all, I would like to give my special and sincere thanks to my advisor, Dr. Ing-Ray Chen, who has constantly provided valuable guidance and patience to my research work. I always appreciate that Dr. Chen accepted me as his Ph.D. student. With his help, I could survive as a Ph.D. student.

I would also like to thank Professors Denis Gracanin, Chang-Tien Lu, Hazhir Rahmandad, and Naren Ramakrishnan for serving on my thesis committee and for their valuable advice and comments.

If I were to name a person who deserves the most thanks, it would be my wife Kun Hu. I would like to thank her for her love and support. Her presence, at happy and hard times, and invaluable moral help were instrumental to the conduct of my research. I thank her for all her patience and support. I am indebted to my mother Xiping Zhang and my father Hehe Li for their constant love, encouragement, and support. I also offer my thanks to my lovely daughter Zifei Li, for the joy she brings to our family.
Contents

1 Introduction .................................................. 1
  1.1 Wireless Mesh Networks ..................................... 1
  1.2 Motivation and Research Statement ............................. 2
  1.3 Research Contributions ....................................... 4
    1.3.1 Per-User based Mobility Management ......................... 4
    1.3.2 Integrated Mobility and Cache Consistency Management for Mobile Data Access ................................. 6
    1.3.3 Integrated Mobility and Service Management for Mobile Multicast ........................................... 7
    1.3.4 Model-based Analysis and Performance Optimization ......................................................... 8
    1.3.5 Integrated Mobility and Service Management for Mobile Ad Hoc Networks .......................... 10
  1.4 Dissertation Organization ..................................... 11

2 Related Work ................................................ 12
  2.1 Mobility Management ....................................... 12
    2.1.1 Mobility Management in Cellular Networks ................ 12
    2.1.2 Mobility Management in Mobile IP Networks ................ 16
    2.1.3 Mobility Management in Mobile Ad Hoc Networks .............. 19
    2.1.4 Mobility Management in Wireless Mesh Networks ............ 22
  2.2 Cache Consistency Management and Mobile Data Access .......... 24
    2.2.1 Cache Invalidation ........................................... 24
    2.2.2 Non-Cache-Invalidation based Approaches ..................... 26
  2.3 Multicast and Supporting Algorithms ......................... 27
    2.3.1 Multicast in Mobile IP Networks ............................. 27
    2.3.2 Multicast in Wireless Mesh Networks .......................... 30
  2.4 Secure and Reliable Multicast and Supporting Algorithms ........ 32
3 System Models and Assumptions

4 Mobility Management based on Per-User Pointer Forwarding

4.1 Static Anchor Scheme

4.1.1 Location Handoff

4.1.2 Service Delivery

4.2 Dynamic Anchor Scheme

4.2.1 Service Delivery

4.3 Performance Modeling

4.3.1 Static Anchor Scheme

4.3.2 Dynamic Anchor Scheme

4.3.3 Parameterization

4.3.4 Performance Metrics

4.4 Performance Analysis and Numerical Results

4.4.1 Proposed Pointer Forwarding Schemes

4.4.2 Proposed Schemes vs. Baseline Schemes

4.4.3 The WMM Scheme

4.4.4 Proposed Schemes vs. The WMM Scheme

4.4.5 Sensitivity Analysis

4.5 Summary

5 Routing-based Mobility Management with Pointer Forwarding

5.1 LMMesh

5.1.1 Routing-based Location Update and Pointer Forwarding

5.1.2 Integration and Its Impact

5.1.3 Location Search Procedure

5.1.4 Data Packet Routing

5.1.5 Multiple Gateways

5.1.6 Scalability of LMMesh

5.2 Performance Model

5.2.1 Analytical Model for LMMesh

5.2.2 Parameterization

5.2.3 Performance Metrics
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.3 Performance Analysis</td>
<td>80</td>
</tr>
<tr>
<td>5.3.1 Performance Evaluation of LMMesh</td>
<td>81</td>
</tr>
<tr>
<td>5.3.2 Performance Comparison</td>
<td>83</td>
</tr>
<tr>
<td>5.3.3 Sensitivity Analysis</td>
<td>89</td>
</tr>
<tr>
<td>5.4 Summary</td>
<td>90</td>
</tr>
<tr>
<td>6 Integrated Mobility and Cache Consistency Management for Mobile Data Access</td>
<td>91</td>
</tr>
<tr>
<td>6.1 Adaptive Per-User Per-Object Cache Consistency Management</td>
<td>92</td>
</tr>
<tr>
<td>6.1.1 DPM versus CCM</td>
<td>92</td>
</tr>
<tr>
<td>6.1.2 Data Access and Caching</td>
<td>93</td>
</tr>
<tr>
<td>6.1.3 MC Mobility and Data Migration</td>
<td>95</td>
</tr>
<tr>
<td>6.2 Analytical Modeling</td>
<td>97</td>
</tr>
<tr>
<td>6.2.1 SPN Models for APPCCM</td>
<td>97</td>
</tr>
<tr>
<td>6.2.2 Parameterization</td>
<td>101</td>
</tr>
<tr>
<td>6.2.3 Performance Metrics</td>
<td>102</td>
</tr>
<tr>
<td>6.3 Performance Analysis and Numerical Results</td>
<td>106</td>
</tr>
<tr>
<td>6.3.1 Performance Evaluation of APPCCM</td>
<td>107</td>
</tr>
<tr>
<td>6.3.2 Comparative Performance Study</td>
<td>110</td>
</tr>
<tr>
<td>6.3.3 Simulation Validation</td>
<td>112</td>
</tr>
<tr>
<td>6.4 Summary</td>
<td>114</td>
</tr>
<tr>
<td>7 Integrated Mobility and Service Management for Mobile Multicast</td>
<td>116</td>
</tr>
<tr>
<td>7.1 Dynamic Agent-based Hierarchical Multicast</td>
<td>117</td>
</tr>
<tr>
<td>7.1.1 Overview</td>
<td>117</td>
</tr>
<tr>
<td>7.1.2 Member Join and Leave</td>
<td>119</td>
</tr>
<tr>
<td>7.1.3 Mobility Management and Tree Maintenance</td>
<td>121</td>
</tr>
<tr>
<td>7.1.4 Multicast Packet Delivery</td>
<td>122</td>
</tr>
<tr>
<td>7.2 Performance Model</td>
<td>123</td>
</tr>
<tr>
<td>7.2.1 Stochastic Petri Net (SPN) Model</td>
<td>124</td>
</tr>
<tr>
<td>7.2.2 Performance Metrics</td>
<td>127</td>
</tr>
<tr>
<td>7.3 Performance Analysis and Numerical Results</td>
<td>130</td>
</tr>
<tr>
<td>7.3.1 Performance Evaluation</td>
<td>131</td>
</tr>
</tbody>
</table>
7.3.2 Comparative Performance Study ........................................... 133
7.3.3 Simulation Validation .......................................................... 137
7.3.4 Discussion ........................................................................ 139
7.4 Summary ............................................................................. 139

8 Integrated Mobility and Service Management for Reliable and Secure Mobile Multicast

  8.1 Hierarchical Agent-based Secure and Reliable Multicast ................................. 143
    8.1.1 Security Key Management ....................................................... 143
    8.1.2 Reliable Multicast Data Delivery .............................................. 144
    8.1.3 Dynamic Group Membership Management .................................. 147
    8.1.4 Mobility Management ............................................................ 147
  8.2 Performance Model .................................................................. 148
  8.3 Performance Evaluation ............................................................ 155
  8.4 Comparative Performance Study ................................................... 159
  8.5 Simulation Validation ................................................................ 163
  8.6 Summary ................................................................................ 163

9 Integrated Mobility and Service Management in Mobile Ad Hoc Networks 166

  9.1 Dual-Region Mobility Management for Location-based Routing ...................... 167
    9.1.1 Overview ............................................................................ 167
    9.1.2 Location Update ................................................................... 169
    9.1.3 Location Query ..................................................................... 172
    9.1.4 Data Packet Delivery .............................................................. 174
    9.1.5 Maintenance of Home Region ................................................ 174
    9.1.6 Application of Optimal Setting ............................................... 175
  9.2 Mathematical Model for DrMoM .................................................... 176
    9.2.1 Assumptions ....................................................................... 176
    9.2.2 Mathematical Model ............................................................... 177
    9.2.3 Home Region Maintenance Cost $C_m$ ..................................... 180
    9.2.4 Total Communication Cost $C$ ................................................ 181
  9.3 Performance Evaluation .................................................................. 181
    9.3.1 Performance Comparison ....................................................... 182
List of Figures

1.1 The architecture of a typical wireless mesh network. ............................. 2

4.1 The handling of location handoffs in the proposed pointer forwarding schemes (LocUpdate means a location update message, and PF means pointer forwarding). ........ 41

4.2 The location search procedure for newly arrived Internet sessions in the dynamic anchor scheme. ................................................................. 43

4.3 The location search procedure for newly arrived Intranet sessions in the dynamic anchor scheme. ................................................................. 45

4.4 The SPN model for the static anchor scheme. ................................. 45

4.5 The SPN model for the dynamic anchor scheme. ................................. 47

4.6 Total communication cost vs. K. .................................................. 53

4.7 $C_{static}$ – $C_{dynamic}$ vs. SMR. ............................................... 54

4.8 Optimal K vs. SMR. ................................................................. 54

4.9 Cost difference vs. SMR between the proposed schemes and baseline schemes. ...... 55

4.10 The SPN model for the WMM scheme. ........................................... 57

4.11 Total communication cost vs. $\zeta$ in the WMM scheme. ...................... 60

4.12 $\zeta_c$ vs. SMR. ................................................................. 61

4.13 Proposed schemes vs. baseline schemes, under two different combinations of $\alpha$ and $\beta$. 62

4.14 Total communication cost vs. K, assuming the hexagonal network coverage model. 63

5.1 The pointer forwarding method (LocUpdate means a location update message, and $FP$ means a forwarding pointer). ............................... 66

5.2 The location search procedure for Internet sessions. ............................ 69

5.3 The location search procedure for intranet sessions. ............................ 70

5.4 The SPN model for LMMesh. ..................................................... 74

5.5 Cost vs. K. ........................................................................... 81
5.6 \( K_{\text{optimal}} \) vs. \( SMR \) .................................................. 81
5.7 Cost vs. \( \zeta \) ................................................................. 82
5.8 Average forwarding chain length vs. \( \zeta \) ......................... 82
5.9 Cost vs. \( \omega_w/\omega_s \) ....................................................... 83
5.10 \( K_{\text{optimal}} \) vs. \( \omega_w/\omega_s \) ............................................ 83
5.11 Performance comparison: cost vs. \( SMR \) ............................. 86
5.12 Performance comparison: average forwarding chain length vs. \( SMR \) ................................. 86
5.13 Performance comparison: cost vs. \( \zeta \) ................................. 87
5.14 Performance comparison: average forwarding chain length vs. \( \zeta \) ................................. 87
5.15 Performance comparison: cost vs. \( \omega_w/\omega_s \) .................. 88
5.16 Performance comparison: cost vs. \( \lambda_I \) .......................... 88
5.17 Performance comparison: cost vs. \( \mu_I \) ............................ 89
5.18 Cost vs. \( K \) under the hexagonal network coverage model ... 89
5.19 Performance comparison: cost vs. \( SMR \) under the hexagonal network coverage model 90

6.1 Examples of query and update processing in APPCCM ......................... 95
6.2 Threshold-based data object migration in DPM .................................. 96
6.3 Pointer forwarding based location management in CCM .......................... 96
6.4 The SPN model for DPM ....................................................... 98
6.5 The SPN model for CCM ....................................................... 100
6.6 Cost vs. \( K/L \) (QUR = 0.5; QMR = 8.0) .................................. 107
6.7 Cost vs. \( K/L \) (QUR = 2.0; QMR = 8.0) .................................. 107
6.8 Cost vs. \( \text{QUR} \) ............................................................... 108
6.9 The optimal threshold \( K_{\text{optimal}} \) vs. \( \text{QUR} \) in DPM ....................... 108
6.10 Cost vs. \( \text{QMR} \) ............................................................. 109
6.11 Cost vs. \( P_{\text{active}} \) under two different values of \( \text{QUR} \) .............. 109
6.12 Cost vs. \( \text{QUR} \) under different combinations of \( \alpha \) and \( \beta \) ...... 110
6.13 Performance comparison: cost vs. \( \text{QUR} \) .................................. 111
6.14 Cost vs. \( \text{QMR} \) ............................................................. 111
6.15 Simulation validation: cost vs. \( K/L \) (QUR = 0.5) ....................... 113
6.16 Simulation validation: cost vs. \( K/L \) (QUR = 2.0) ....................... 113
6.17 Simulation validation: cost vs. \( \text{QUR} \) .................................. 113
List of Tables

3.1 Common input parameters. .......................................................... 38

4.1 The parameters and notations used in performance modeling and analysis. ........ 44
4.2 The interpretation of places and transitions defined in the SPN model for the static
anchor scheme. .............................................................................. 46
4.3 The interpretation of additional places and transitions defined in the SPN model for
the dynamic anchor scheme. .......................................................... 48
4.4 The parameters and their default values used. ............................................ 53

5.1 Parameters and their physical meanings used in performance modeling and analysis. 73
5.2 Parameters and their default values. ....................................................... 81

6.1 Fields of a caching status table entry. ................................................... 93
6.2 Parameters used in performance modeling and analysis. ............................... 97
6.3 The meanings of places and transitions defined in the SPN model for DPM. ........ 98
6.4 The meanings of places and transitions defined in the SPN model for CCM. ........ 100
6.5 Parameters and their values used in performance analysis. .......................... 106

7.1 Parameters and notations used in performance modeling and analysis. .............. 123
7.2 The meanings of places and transitions defined in the SPN model for DAHM. ....... 126
7.3 Parameters and their typical values used in performance evaluation. ................. 130

8.1 Security keys used in HASRM. ......................................................... 144
8.2 Parameters and their physical meanings. ............................................... 149
8.3 The meanings of places and transitions defined in the SPN model for HASRM. .... 152
8.4 Parameters and their typical values. ..................................................... 155

9.1 The notations used in this paper. ......................................................... 170
9.2 The notations used in performance analysis. .............................................. 176

10.1 Computational time (in seconds) needed to find a solution by various search methods
(BF stands for Brute-Force). ................................................................. 188

10.2 The optimal values of $R_l$ and $R_h$ found by various search methods (BF stands for
Brute-Force). ...................................................................................... 189

10.3 The minimum communication cost found by various search methods (BF stands for
Brute-Force). ...................................................................................... 190
Chapter 1

Introduction

1.1 Wireless Mesh Networks

Wireless mesh networking is a cost-effective solution for providing last-mile community-based broadband Internet access services. The typical usage scenarios of a wireless mesh network (WMN) include home networking, community networking, enterprise networking, transportation systems networking (e.g., bus and train), and health/medical systems networking [1]. A WMN usually has a small to medium size in these scenarios. A WMN consists of two types of network components: wireless mesh routers and mesh clients [1]. A mesh router (MR) is functionally similar to a wired router, i.e., it routes IP packets at the network layer. However, MRs communicate among each other via one or multiple broadband wireless interfaces. The group of MRs forms a mesh networking infrastructure called a wireless mesh backbone that routes packets and provides last-mile broadband Internet connectivity to mesh clients. Because MRs are typically static and has minimal mobility, the wireless mesh backbone has a flexible yet static topological structure. A mesh client (MC) represents an end-user device with wireless access capability, and unlike MRs, MCs are usually mobile and may change its location and serving MR frequently during its stay in a WMN. The serving MR of an MC is the MR to which it is directly connected. A WMN is seamlessly interconnected to the Internet through gateways, which can also be used to integrate a WMN with existing wireless networks, such as Cellular Networks, Mobile Ad hoc Networks (MANETs), and Wireless Sensor Networks (WSNs). Generally, one or more MRs in a WMN serve as the Internet gateways and route network traffic originated from or destined to the Internet.

The architecture of a typical WMN is presented in Fig. 1.1. As the figure illustrates, the wireless mesh backbone serves as both an infrastructure that integrates various types of networks and an access point that expands the Internet to last-mile wireless mobile end-users within a
community. Because the wireless mesh backbone consists of MRs connected via wireless links, the network topology and structure of a WMN can be very flexible, making it an attractive solution for providing last-mile broadband Internet access services in a diverse set of configurations. The up-front cost of network deployment and the maintenance cost are also greatly reduced due to this flexibility. Therefore, two key advantages of WMNs are great flexibility and cost effectiveness. The flexibility of the mesh structure also leads to other features that are attractive to both the industry and academia, e.g., self organization, self configuration, easiness of deployment and management, flexible community-based coverage, and fault tolerance. Over the past years, we have seen a trend of increasing deployment of WMNs and interest from the industry as more mesh networking products emerge in the market.

1.2 Motivation and Research Statement

This dissertation focuses on protocol design, which is significantly affected by network technologies. WMNs share some characteristics with other types of wireless networks, such as wireless cellular networks and MANETs. On the other hand, WMNs have unique characteristics that call for specific protocol designs to optimize application performance.
Wireless cellular networks have a hierarchical structure organized as registration areas connected by mobile switch centers. Each registration area covers a group of base stations (cells) and hosts a location database that is responsible for storing the location information of mobile terminals. A hierarchical HLR-VLR scheme (Ref. Chapter 2) is employed to track the location information of mobile users and route data packets. This hierarchical structure allows wireless cellular networks to span a large area and serve a huge number of mobile users. In contrast, a WMN is normally used as a last-mile network connecting to the Internet and typically has a flat mesh structure, consisting of many MRs of similar capability connected via wireless broadband links. Such a flat mesh structure means that protocol designs should aim to make a WMN exhibit self-organization, self-configuration, and fault tolerance properties. Because there is no natural analogy of registration areas and HLR/VLR in WMNs, existing protocols proposed for wireless cellular networks that rely on hierarchical network structures and the existence of dedicated location servers are not appropriate for WMNs. Additionally, because communication links between MRs and between an MR and an MC are wireless, protocol designs proposed for WMNs must take advantage of wireless broadcast.

A MANET, like a WMN, has a flat structure consisting of mobile nodes that act as routers and communicate through wireless links. However, these “mobile” routers in MANETs typically are running on battery as opposed to “static” MRs in WMNs running on permanent external power sources. This means that power consumption is a critical metric for protocol design for MANETs but not necessarily true for WMNs. Consequently, an objective of protocol design for MANETs is to optimize the computational efficiency so as to maximize the battery life. Protocol design for WMNs, on the other hand, is more about maximizing the network capacity and achievable throughput. Additionally, because a MANET does not have a static routing topology, protocol design must consider intermittent connectivity between nodes. As a result, routing protocols developed for MANETs, such as Ad hoc On-demand Distance Vector (AODV) and Optimized Link State Routing (OLSR), must address route discovery and route maintenance issues. Route discovery and maintenance is not necessary in WMNs because as long as the serving MR of the destination is node is known (a problem addressed by mobility management), the static mesh backbone knows how to route data packets.

In this dissertation research, we design and analyze several mobility management protocols with the objective of minimizing the overall network cost, and develop the notion of integrated mobility and service management for key applications in WMNs, including mobile data access and mobile
multicast. More specifically, we first investigate the problem of mobility management in WMNs for which we propose three efficient schemes (Chapters 4 and 5) that minimize the overall network communication cost incurred collectively by mobility management and packet delivery. Mobility management is critical for the proper operation of WMNs and is the basis for uninterrupted network services because it maintains necessary information for a service to be delivered to a mesh client that changes its location frequently. We further study per-user based integrated mobility and service management for which we propose protocols that support efficient mobile data access services with cache consistency management (Chapter 6), and mobile multicast services (Chapters 7 and 8). The most salient feature of our protocols is that they are optimal on a per-user basis (or on a per-group basis for mobile multicast), that is, the overall network communication cost incurred is minimized for each individual user (or group). Per-user based optimization is critical because mobile users normally have vastly different mobility and service characteristics. Thus, the overall cost saving due to per-user based optimization will be cumulatively significant with an increasing mobile user population. We also investigate extending the design notion of integrated mobility and service management to MANETs and propose a scalable mobility management scheme for location-based routing in MANETs (Chapter 9) based on the design notion.

1.3 Research Contributions

1.3.1 Per-User based Mobility Management

A major use of WMNs is as a wireless backbone for providing last-mile broadband Internet access services to MCs in a multi-hop way, through the gateway that is connected to the Internet [3]. Because MCs may change their locations and serving MR frequently during their stay within a WMN, mobility management is critical for the proper operation of the WMN. Specifically, it needs to be guaranteed that services are delivered to MCs continuously without interruption even when they are roaming around in the WMN. Mobility management consists of location management and handoff management [4]. Location management keeps track of the location information of mobile hosts, through location registration and location update operations. Handoff management maintains ongoing connections of mobile hosts while they are moving around and changing their locations in the network. In this dissertation research, we will focus on location management.

As the MANET technology is new in recent years, there is not much research on mobility management [6]. The first major contribution of the dissertation research is that we propose and analyze two per-user based mobility management schemes for WMNs, namely, the static anchor
scheme and dynamic anchor scheme. The static anchor and dynamic anchor schemes are based on pointer forwarding [13]. The basic idea of pointer forwarding approaches is that a chain of forwarding pointers is maintained to track an MC’s current location as the MC moves and changes its serving MR, and to deliver services to the MC following the chain. A pointer is an entry in the routing table that points to the next MR on the chain. The length of the forwarding chain is a key factor influencing the network communication cost incurred by the MC for mobility management and service delivery collectively. We use a threshold for the forwarding chain length as a key parameter to regulate the allowable forwarding chain length and consequently to tune the performance of the schemes. The optimal threshold for the forwarding chain length is determined for each individual MC dynamically based on the MC’s specific mobility and service characteristics. We identify optimal design settings under which the static anchor and dynamic anchor schemes are optimized. We also demonstrate through comparative performance analysis that our mobility protocols outperform existing protocols including WMM [6].

Based on the work above, we further investigate how caching location information of MCs on MRs can improve the performance of our mobility management schemes. We propose LMMesh, a per-user routing-based mobility management scheme with pointer forwarding for WMNs. LMMesh integrates routing-based location update and pointer forwarding into a single scheme that exploits the advantages of both methods, while avoiding their drawbacks. A well-known advantage of routing-based location update is that it enables the propagation of location information of MCs to the concerning parties using regular data packets originated from the MC, therefore avoiding the signaling overhead of explicit location update messages. Routing-based location update, however, does not work well for MCs that do not have active network sessions or MCs that are not sending data packets. Pointer forwarding is a solution for location management that uses explicit location update messages. It works for those MCs for which routing based location update does not work well, at the expense of additional signaling cost for the location update messages.

Although routing-based location update and pointer forwarding have been individually applied in many mobile communication networking studies, the integration of them and the impact of this integration on the overall network performance has not been studied. The major contribution of this work is that we formulate the interaction between routing-based location update and pointer forwarding and analyze the impact of this integration on the overall network communication cost incurred. For this work, we also eliminate the assumption that WMNs have a single gateway, making LMMesh a generic scheme that works for both single-gateway and multi-gateway WMNs.
1.3.2 Integrated Mobility and Cache Consistency Management for Mobile Data Access

The second major contribution of the dissertation research is that we develop the notion of integrated mobility and service management for key applications in WMNs. The first application we investigate is mobile data access services. Mobile data access, which is fundamental to client-server computing in mobile environments [50], is challenging due to mobility and resource constraints of clients, and bandwidth constraints of wireless communications [51,52]. An additional challenge as a result of these characteristics is that mobile data access must cope with voluntary or involuntary disconnection of mobile clients. That is, mobile data access must support disconnected operations [53]. Caching is a key technique for improving the performance of mobile data access because it alleviates constraints such as highly variable mobile connectivity and wireless bandwidth limitations, and it significantly reduces the latency for answering a query. Caching is also the basis for disconnected operations in mobile data access.

Data caching is particularly beneficial for mobile data access in WMNs for a major reason. Specifically, because a WMN is a cost-effective solution for last-mile broadband Internet access, Internet traffic due to mobile data access must pass through the gateway and is expected to dominate network traffic in WMNs. The implication of this observation is that the gateway is potentially the bottleneck under heavy Internet traffic and network congestion is highly probably to happen around the gateway. Caching is an efficient solution for mitigating the problem because it can significantly reduce the number of uplink and downlink messages passing through the gateway due to client-server mobile data access, thereby mitigating the performance bottleneck at the gateways in WMNs [54].

A central issue closely related to caching in mobile data access is cache consistency management because clients may not be able to keep cached data synchronized due to mobility and disconnection. In this dissertation research, we propose and analyze an Adaptive Per-user Per-object Cache Consistency Management (APPCCM) scheme for mobile data access in WMNs. APPCCM supports strong data consistency semantics through integrated cache consistency and mobility management. The objective of APPCCM is to minimize the overall network cost incurred collectively by data query/update processing, cache consistency management, and mobility management. APPCCM provides two data access and caching modes: a data object can be cached either directly at the MC, or at a data proxy running on an MR dynamically selected by APPCCM.

APPCCM is adaptive, per-user and per-object because for each individual MC, the decision
of where to cache accessed data objects is made dynamically and independently for each object based on the MC’s mobility and data query/update characteristics, and the WMN conditions. We develop a computational procedure for dynamically calculating the overall communication cost in APPCCM, given parameters characterizing the MC’s mobility and data query/update characteristics, and the WMN conditions.

1.3.3 Integrated Mobility and Service Management for Mobile Multicast

The third major contribution of the dissertation research is the design notion of integrated mobility and service management for mobile multicast applications in WMNs. Due to the broadcasting nature of wireless communications and the community-oriented nature of WMNs, group communication based on multicast routing is expected to be a common communication paradigm in WMNs. For example, many popular network applications today are based on the single-source group communication paradigm, and require efficient delivery of various types of contents, e.g., weather forecasts, stock prices, news, and real-time audio/video streams, from a single source to a group of mobile users in wireless networks, and particularly, in WMNs. These applications are multicasting in nature, and can therefore be efficiently implemented by a multicast algorithm. The (potentially high) mobility of multicast group members, however, poses a challenge to the design and development of efficient multicast algorithms in WMNs. More specifically, the multicast algorithm must efficiently support user mobility such that group members can continue to receive subscribed multicast contents and data when they move and change their serving MRs frequently.

In this dissertation research, we propose and analyze a mobile multicast algorithm for WMNs named Dynamic Agent-based Hierarchical Multicast (DAHM) that supports potentially highly mobile users and dynamic multicast group membership. The objective of DAHM is to minimize the overall network communication cost incurred by multicast packet delivery, mobility management, and multicast tree maintenance. DAHM dynamically selects Multicast Agents (MAs) [72] running on MRs for integrated mobility and multicast service management, and combines backbone multicast routing and local unicast routing into an integrated algorithm for efficient multicast packet delivery. As the name suggests, DAHM employs a dynamic two-level hierarchical multicast structure. At the upper level is a dynamic shortest-path multicast tree whose nodes are MRs, and whose leaves are dynamically selected MAs for integrated mobility and multicast service management. The dynamic multicast tree at the upper level forms a multicast backbone, as multicast packets are first forwarded from the gateway to all the MAs through the tree. The multicast tree is
updated to maintain its structural properties every time an MA joins or leaves due to user mobility and group membership changes.

An MA runs on an MR and acts as a point of regional registration for integrated mobility and multicast service management. Each MA and the multicast group members it currently services form a local multicast group at the lower level of the hierarchy. The MA delivers multicast packets to each group member individually via unicast routing. Therefore, the proposed multicast algorithm features an integrated design that operates in multicast routing mode within the multicast backbone, and in unicast routing mode within each local multicast group. The service region size of an MA is a key parameter controlling the tradeoff between the communication cost incurred at the upper level and that incurred at the lower level. We develop a mathematical model in the dissertation research to determine the optimal service region size that minimizes the overall communication cost.

Based on the idea of integrated mobility and service management for mobile multicast, we extend the design of DAHM with support for reliable and secure mobile multicast, and propose an algorithm called Hierarchical Agent-based Secure and Reliable Multicast (HASRM). HASRM employs the same two-level hierarchical multicast structure as in DAHM and utilizes MAs for mobility management, security key management, dynamic group membership management, and reliable multicast data delivery, in an integrated manner. Like DAHM, HASRM dynamically determines the optimal service region size of an MA that minimizes the total communication cost incurred, by balancing the tradeoff between the communication cost incurred at the upper level and that incurred at the lower level.

1.3.4 Model-based Analysis and Performance Optimization

The forth major contribution is the development of a unified modeling and analysis framework for performance evaluation of mobility and service management protocols for WMNs. The primary objective of this dissertation research is to design and analyze enabling mobility management protocols, and based on which, integrated mobility and service management protocols that optimize mobile network services in WMNs. We explore the design space and identify key design choices that optimize the performance of these integrated mobility and service management protocols. Specific to performance optimization, the central objective is to minimize the overall network communication cost incurred per time unit by each protocol on a per-user basis. We emphasize cost minimization because the network cost of a scheme has a significant impact on network and
protocol performance, characterized by the following metrics, including throughput, latency, power consumption, packet delivery ratio, response time, etc. Our methodology for identifying the optimal design choices for network cost minimization is model-based analysis, using a unified modeling framework, which generally consists of the following components:

- An analytical performance model that captures the essential of the protocol;
- A set of parameters describing the mobility and service characteristics and usage practices of mesh clients, as well as the properties and conditions of the WMN;
- A cost metric for performance analysis (the overall network communication cost incurred per time unit);
- A computational procedure for calculating the cost.

Model-based analysis in this context refers to performance analysis using a mathematical model that calculates the overall network communication cost incurred by a protocol. We use Stochastic Petri Net (SPN) techniques as a primary tool for constructing mathematical models for most of the protocols proposed in this dissertation research. An SPN model is essentially a concise representation of a continuous-time semi-Markov (CTSM) model, allowing states, state transitions, and the rates at which state transitions occur due to mobility and service events to be defined. The end product is a CTSM state machine describing the behavior of a mesh client as it migrates from one state to another as a result of triggering mobility and service operations from executing our proposed integrated mobility and service protocol. By solving the CTSM to get the probability that the mesh client is in a particular state, and by associating a cost to each state of the CTSM, we can calculate the expected cost for integrated mobility and service management. Therefore, we use SPN models as mathematical models for performance analysis. We also use queuing theory as a auxiliary tool for performance modeling and analysis.

The unified modeling and analysis framework proposed is generic, allowing integrated mobility and service management protocols to be analyzed and optimal design choices to be identified for cost minimization. We utilize the framework for performance comparison of our proposed integrated mobility and service management protocols against existing protocols in the literature. The validity of our model-based analysis methodology is ascertained through simulation validation.
1.3.5 Integrated Mobility and Service Management for Mobile Ad Hoc Networks

The last but not least contribution of this dissertation research is extending the design notion of integrated mobility and service management to MANETs. Like WMNs, a MANET is also a self-organizing self-configuring network that dynamically maintains a mobile routing backbone composed of mobile nodes connected by wireless links for multi-hop routing. Therefore, WMNs and MANETs share some common network characteristics, and it is potentially beneficial to apply the design notion to MANETs.

In the literature the prevalent location service in MANETs is hashing-based with which each mobile node is assigned a home region through hashing [108,110,112,113,116]. The nodes in the home region serve as location servers for that mobile node. A mobile node sends location updates to its location servers when it moves. To locate a destination node, a source node sends a location query to the destination node’s location servers. Although a hashing-based location service is highly scalable, it has a major drawback: a source node has to contact the location servers of the destination node regardless of how close it is away from the destination node. If the two nodes are close to each other, contacting the location servers which may be far away geographically incurs unnecessary overhead. One way to solve this problem is to have a mobile node periodically exchange up-to-date location information with neighboring nodes in a local region [111,117]. If some node in the local region of the source node knows the location of the destination node, the source node can locate the destination node utilizing only local location information from the neighboring nodes, without having to query the destination node’s home region. It is also possible that the source node is within the local region of the destination node and therefore knows where the destination node is located using only local location information it keeps.

In the dissertation research, we propose and analyze an integrated mobility and service management scheme for location-based routing in MANETs called Dual-region Mobility Management (DrMoM) based on the idea of employing local regions to complement existing home region based location service schemes that assign home regions to mobile nodes and have mobile nodes in the home region of a mobile node serve as location servers for that node. Relative to existing work utilizing home region based location service [108,110,112,113,116] and local region based location service [111,117], our contribution is to dynamically determine the optimal home region size and local region size for each mobile node based on the mobile node’s runtime mobility and service characteristics to minimize network cost. We perform a comparative analysis of DrMoM against...
a conventional location-based routing protocol called SLURP [113] that handles mobility management using static home regions. We demonstrate that DrMoM under optimal settings outperforms SLURP in terms of the overall network cost incurred.

1.4 Dissertation Organization

This dissertation is organized as follows. Chapter 2 surveys existing work and achievements in relevant research areas and contrast our work with existing work. The system models and assumptions made in this dissertation research are presented in Chapter 3. Chapter 4 through Chapter 9 comprise the main content of this dissertation. Specifically, Chapter 4 presents two mobility management schemes for WMNs based on pointer forwarding, namely, the static anchor and dynamic anchor schemes. Chapter 5 introduces LMMesh, the mobility management scheme that integrates routing-based location update and pointer forwarding. The APPCCM scheme proposed for cache consistency management in mobile data access in WMNs is presented in Chapter 6. Chapter 7 presents the DAHM algorithm for supporting efficient mobile multicast in WMNs. The HASRM algorithm proposed for secure and reliable mobile multicast in WMNs is introduced in Chapter 8. Chapter 9 presents the DrMoM scheme proposed for integrated mobility and service management in MANETs. Issues and solutions related to the applicability and implementation of the protocols proposed in this dissertation are discussed in Chapter 10. The dissertation concludes with Chapter 11.
Chapter 2

Related Work

2.1 Mobility Management

Mobility management has been studied intensively for cellular networks and Mobile IP networks. A variety of mobility management schemes and protocols have been proposed over the past years. A comprehensive survey of mobility management for cellular networks and Mobile IP networks can be found in [4] and [5]. For wireless mesh networks, mobility management is relatively unexplored [6].

2.1.1 Mobility Management in Cellular Networks

Baseline Schemes

There are two commonly used standard schemes for mobility management in cellular networks [4], namely, the Electronic and Telephone Industry Associations (EIA/TIA) Interim Standard 41 (IS-41) [36], and the Global System for Mobile Communications (GSM) Mobile Application Part (MAP) [37]. IS-41 is widely used in North America, whereas GSM MAP is typically used in Europe. Both standards employ a two-level hierarchical database structure for mobility management. The service coverage area of a cellular network is divided into Registration Area (RA), each of which covers a group of base stations (cells) connected to the same Mobile Switch Center (MSC). There is a location database located at the lower level of the database hierarchy called a Visitor Location Register (VLR) in each RA that is responsible for registering the location information of Mobile Terminals (MTs) currently within the RA. Each MT is always associated with a VLR and the VLR of the MT is dynamically changed while the MT is roaming in the network. Each MT is also permanently registered with a location database called a Home Location Register (HLR) located at the upper level of the database hierarchy. Information about an MT including the types of services the MT subscribes and its location information is stored as a user profile in the HLR of the MT. Specifically, the location information of an MT stored in its HLR specifies the VLR with which the
MT is currently associated.

While roaming in a cellular networks, an MT performs a "location registration (location update)" procedure to update its location information with the serving VLR and HLR. When the MT moves across cells within the same RA, it sends a location registration message to the VLR of the current RA to update its location information stored in the VLR, i.e., the ID of the cell in which it is located. When the MT moves across the boundary from one RA to another, it is registered and associated with the new VLR in the new RA, and the new VLR sends a location registration message to its HLR to update its location information stored in the HLR, i.e., the ID of the new VLR. The HLR updates the user profile of the MT and sends a registration acknowledgment message to the new VLR, informing it that the location registration is completed. The HLR also sends a registration cancellation message to the old VLR of the MT. The old VLR removes the record of the MT and sends a cancellation acknowledgment message back to the HLR. To deliver a call to an MT, the current location of the called MT must be known to the calling MT. The location of the MT is determined in a hierarchical way through the HLR of the MT, the current VLR with which the MT is associated, and finally the cell in which the MT is located. After the MT is located, a connection between the calling MT and the called MT can be setup.

A significant problem with the standard HLR-VLR scheme is that the signaling traffic generated by location management and call delivery can be considerably high for a large number of MTs, especially if the MTs have high mobility. To solve the problem, many schemes with optimization considerations have been proposed for reducing the signaling traffic.

**Per-user Location Caching**

The per-user location caching strategy [38] aims to reduce the volume of signaling and database access traffic for locating an MT when a call to the MT needs to be delivered. The basic idea of per-user location caching is to store the location information of a called MT in a cache maintained in the cell or RA in which the calling MT resides. When a call from the calling MT is initiated for the called MT, the cache is first checked to see if an entry for the called MT exists. If no entry is found, the standard HLR-VLR scheme is followed to deliver the call. Otherwise, the VLR of the called MT specified in the cache entry is queried to locate the MT. There is a cache hit if the MT is still associated with the VLR. A cache miss occurs if the MT has already moved to another RA. In a cache hit, the MT is located, whereas in a cache miss, the standard HLR-VLR scheme is used to locate the MT.
Because the location information of an MT kept in a location cache becomes obsolete when the MT moves to a different location, methods for location cache maintenance are needed. There are two methods for maintaining a location cache, namely, lazy cache maintenance and eager cache maintenance. In lazy cache maintenance, the cached location information of an MT is updated on every location search (paging) for locating the MT, whereas in eager cache maintenance, the cached location information is updated on every location registration initiated by the MT. Both methods represent a tradeoff between the signaling cost for cache maintenance and the accuracy of cached location information. Lazy cache maintenance reduces the signaling cost for keeping the cached information updated while sacrificing the accuracy of cached location information. Eager cache maintenance maintains relatively high accuracy of cached location information at the expense of increased signaling cost.

**Pointer Forwarding**

The basic idea of the pointer forwarding scheme [13] is to simply set up a forwarding pointer between the old and new VLR when an MT moves from one RA to another, instead of performing a costly location registration with the HLR of the MT, thereby reducing the signaling traffic generated by location management. As the MT moves across RA, a chain of forwarding pointers is setup. The HLR tracks the chain head, i.e., the first VLR of the chain rather than the current serving VLR of the MT. When a call for the MT is to be delivered, the chain head is determined by querying the HLR and the chain is followed to locate the MT. The effect of using the pointer forwarding strategy is that the delay in locating an MT increases as the length of the chain increases. To limit the delay, a threshold $K$ is defined to limit the length of the chain such that the maximum length does not exceed $K$. When the length of the chain reaches $K$ after the MT moves to a new RA, no new forwarding pointer is allowed, instead, a location registration is performed to update the serving VLR of the MT registered in the HLR of the MT. The HLR records the ID of the VLR of the new RA in the entry for the MT. After the location registration, the chain is reset and the new VLR becomes the new chain head. The value of $K$ that effectively reduces the signaling traffic is determined based on the mobility and call arrival characteristics of an MT.

The pointer forwarding scheme can be considered as a variant of a dynamic location update scheme named the movement-based scheme [40]. In the movement-based scheme, an MT tracks the number of RA boundaries it has crossed, i.e., the number of movements, since the last location update. The next location update is performed when the number of movements exceeds a predefined
threshold. Between two consecutive location updates, the MT does not report location changes, as in the pointer forwarding scheme. Therefore, the location information of the MT recorded by its serving VLR is only approximate, and a location search via a terminal paging scheme [4] is needed to locate the MT when an incoming call arrives.

**Local Anchoring**

The local anchoring scheme [39] reduces the signaling traffic generated by location registrations with the HLR by employing a VLR close to an MT as its local anchor for handling location changes of the MT locally. Instead of sending the location registration message to the HLR of the MT every time the MT moves from one RA to another, the message is sent to the local anchor, which is much closer to the MT than the HLR. Therefore, the signaling cost for location registrations is reduced using the local anchoring scheme. The HLR of an MT keeps a pointer to the local anchor of the MT and the pointer is updated when the local anchor of the MT is changed. When a call to an MT arrives, the HLR of the MT is queried to locate its local anchor, which is then queried to locate the current serving VLR of the MT, and finally the current location of the MT is determined such that a connection can be setup.

The local anchoring scheme has two variants that differ in how the local anchor of an MT changes as the MT moves and changes its locations, namely, static and dynamic local anchoring. In both variants, the local anchor of an MT is changed to its current serving VLR when an incoming call to the MT arrives. In dynamic local anchoring, the local anchor of an MT also changes as the MT moves and changes its locations. The decision regarding whether the local anchor of the MT should be changed to the new serving VLR after a movement is made based on the mobility and call arrival characteristics of the MT. Static local anchoring completely eliminates the signaling traffic due to location registrations, while it may cause a significant delay in locating an MT because the MT may be far away from its local anchor if the call arrive rate is low. The delay is bounded in dynamic local anchoring without significantly increasing the signaling traffic incurred by location registrations.

The local anchoring scheme can be considered as a variant of a dynamic location update scheme called the distance-based scheme [40]. In the distance-based scheme, an MT tracks the distance in terms of RA it has moved since the last location update. The next location update is triggered when the distance exceeds a predefined distance threshold. To track the distance it has moved, the MT must have some knowledge about the topology of the network. Location changes of the MT
between two consecutive location updates are not reported. Therefore, like the movement-based scheme, a terminal paging scheme is needed to locate the MT when an incoming call arrives.

### 2.1.2 Mobility Management in Mobile IP Networks

**Mobile IP**

Mobile IP (MIP) is an IETF standard for supporting host mobility in the Internet. MIP [41,42] is the standard mobility management protocol in Mobile IP networks. MIP introduces the concepts of *home networks* and *foreign networks*. Each Mobile Node (MN) is permanently registered with a Home Agent (HA) residing in its home network that acts on behalf of the MN when the MN is in a foreign network. When the MN is in a foreign network, it is temporarily registered with a Foreign Agent (FA), which resides in the foreign network and is responsible for delivering packets to the MN. In MIP, each MN has two IP addresses: a permanent IP address that serves as the identifier of the MN and is known to the Correspondence Node (CN) of the MN, and a Care-of Address (CoA), which is temporarily associated with the MN and changes as the MN changes FA. The CoA tells the current location of the MN, i.e., the FA to which the MN is currently attached. When the MN enters into a new foreign network, it registers with the new FA and obtains a new CoA. The new CoA is registered with the MN’s HA and the location binding information of the MN is updated by the HA. Packet routing to an MN in MIP follows an indirect path because every packet sent from a CN to the MN is first intercepted by its HA and tunneled from the HA to the MN’s current FA using its CoA. The FA finally delivers the packet to the MN. Packet routing in the reverse direction, however, follows a direct path from the MN’s current attachment point to the CN. Therefore, MIP exhibits the so-called *triangular routing* problem.

Route optimization [43] is an enhancement proposed in Mobile IP v6 (MIPv6) [42] that addresses the problem of triangular routing. Route optimization solves the problem by maintaining a direct path between an MN and its CN for packet routing. Therefore, packets originated from the CN are routed directly to the MN without going through the HA. In order to perform route optimization, a CN maintains a binding cache that stores the CoA of the MN with which it communicates. Whenever the CN has a packet to send to an MN, it searches for the CoA of the MN in its binding cache. If an entry is found, the packet is tunneled directly to the FA of the MN specified by the CoA, which then delivers the packet to the MN. If no entry is found, the standard procedure for packet routing in MIP is followed. In the latter case, the CN obtains the CoA of the MN by a *binding update* message from the HA that contains the CoA of the MN, when the HA intercepts the
packet from the CN. Route optimization also addresses the problem of packet loss during handoff from one FA to another. Specifically, when an MN moves to a new foreign network and registers with the new FA, it asks the new FA to send a notification to its old FA such that the old FA is aware of the movement and is able to deliver packets received during the location registration to the HA to the new FA.

Because MIP is a network layer solution for macro-mobility management, it is well suited for global mobility management across both homogeneous and heterogeneous IP-based networks [5]. However, MIP is not an appropriate solution for highly mobile hosts because it incurs considerably large signaling overhead and delay for location update operations when mobile hosts move frequently among subnets. The central problem of MIP is that whenever an MN moves to a new subnet and obtains a new CoA, it needs to register the CoA with its HA, which may be a long distance from the MN’s current location. Therefore, the signaling overhead and delay associated with location update operations in MIP may be significantly large if MNs move and update locations frequently. To remedy the problem, many mobility management schemes that aim to reduce the signaling overhead for location update operations have been proposed. These micro-mobility management schemes can be classified into two categories: tunnel-based and routing-based [5].

**Tunnel-based Micro-Mobility Management**

Mobile IP Regional Registration (MIP-RR) [44] employ Gateway Foreign Agents (GFAs) for regional registrations to reduce the signaling overhead associated with location registrations to the home network. Each GFA in MIP-RR is responsible for local location registrations due to MN movement within a regional network. When an MN first enters into a regional network, it performs a regular location registration with its HA using the address of the GFA of the regional network as the CoA. When the MN moves among subnets and changes FA within the same regional network, it only performs local location registrations with the GFA. When the MN moves across the boundary between two regional networks, it performs a regular location registration to update its GFA with its HA. Because location registrations are performed locally as long as MNs move within their regional networks, the number of location registrations to the home network and therefore the signaling overhead are significantly reduced. Packet delivery in MIP-RR follows the approach in MIP, except that GFAs form an additional layer in the hierarchy that relay packets between HAs and FAs. The idea of employing an additional layer in the hierarchy for localizing location registrations within regional networks can be extended to a more general form that has multiple
levels of hierarchy, as illustrated by Hierarchical Mobile IP (HMIP) [45].

Intra-Domain Mobility management Protocol (IDMP) [46] also features a hierarchical approach to reduce the signaling cost for location registrations and updates. IDMP views the network as a set of mobility domains, each of which consists of a number of subnets. In each mobility domain, there exists a mobility agent that is responsible for local location registrations due to MN movement within the domain. Therefore, a mobility agent is functionally similar to a GFA in MIP-RR. Each MN is assigned two CoAs in IDMP: a Global Care-of Address (GCoA) and a Local Care-of Address (LCoA). The GCoA specifies in which domain the MN currently resides, and the LCoA specifies in which subnet the MN currently is located. When the MN moves among subnets within the same domain, its GCoA remains unchanged, whereas its LCoA is changed and updated with the MA every time it moves from one subnet to another. When the MN moves across the boundary between two mobility domains, it needs to obtain a new GCoA and register this new GCoA with its HA.

Tunnel-based schemes use specialized messages for location registrations and updates. Location registration/update messages are sent whenever an MN changes its locations at the subnet or domain level. Therefore, these location registration/update messages introduce significant signaling overhead, especially for fast moving MNs. Our dynamic anchor and static anchor schemes proposed in Chapter 4 employ optimal per-user pointer forwarding to reduce the signaling overhead as well as to minimize the overall network cost incurred by mobility management and service delivery.

Routing-based Micro-Mobility Management

Cellular IP (CIP) [16] supports MN mobility and fast handoff via regular IP packet routing. In CIP, distributed paging cache and distributed routing cache maintained by routers are used for location management and packet routing, respectively. The distributed paging cache is used to store location information of idle MNs for IP paging, whereas the distributed routing cache is used to store the location information of active MNs and to route IP packets. There is a gateway in each domain that is the hub for packet routing. Packets destined to an MN are first routed to the gateway, which then forwards them to the MN following the host-specific routing path of the MN, as specified by the routing cache entries kept by the routers along the path. The host-specific routing path of the MN is dynamically updated through regular packet routing. Specifically, one of the most distinct characteristics of CIP and generally routing-based schemes is that each data packet originated from an MN carries the up-to-date location information of the sender. For idle
MNs, dummy packets are sent periodically to refresh their location information. Therefore, the location information of an MN kept in a router’s location cache can be dynamically updated when the router processes packets originated from the MN.

Handoff Aware Wireless Access Internet Infrastructure (HAWAII) [17] uses the same idea that host-specific routing paths to MNs for packet routing are dynamically maintained by updating the host-specific routing entries. Unlike CIP, HAWAII uses specialized messages to establish and maintain dynamic host-specific routing paths. There is also a gateway called a domain root router in each domain in HAWAII that is similar to an FA in MIP. When an MN is in a foreign domain, packets destined to it are first intercepted by its HA, which tunnels the packets to the domain root router of the foreign domain. The domain root router further forwards the packets to the MN following its host-specific routing path. When an MN moves from one domain to another, it sends a path setup power-up message to the domain root router of the new domain to initiate the host-specific routing path to the router. Upon receiving the path setup acknowledgment from the domain root router, the MN registers its new CoA with its HA. When the MN moves among subnets within a domain, it uses path setup update messages to establish and maintain its host-specific routing path. The MN also periodically sends path refresh messages to keep its host-specific routing path up-to-date.

Because the update of location information and the maintenance of host-specific routes in pure routing-based schemes solely rely on packet routing, they are essentially opportunistic. Specifically, for idle MNs that is not sending any packets, their location information may become outdated and consequently their host-specific routes may become obsolete. This leads to a major performance deficiency of pure routing-based schemes. The approach adopted by CIP to solve the problem is to send dummy packets at regular intervals. The approach adopted by HAWAII is to use specialized messages to establish and maintain dynamic host-specific routing paths. Both approaches introduce additional overhead that can be significant if MNs moves frequently. In our dissertation research, we plan to investigate how pointer forwarding combined with routing-based approaches can be use to solve the problem with the objective to minimize the overall network traffic collectively incurred by mobility management and packet delivery.

2.1.3 Mobility Management in Mobile Ad Hoc Networks

A Mobile Ad hoc Network (MANET) is a self-organizing and self-configuring network, in which mobile nodes form and maintain a dynamic network topology for multihop routing without a fixed
Routing in a MANET is challenging due to: (1) node mobility that leads to dynamic network topology, and (2) limited resources of mobile nodes such as battery power, processing capability, and storage capacity. In addition, routing protocols for such networks need to be highly scalable because large-scale MANET consisting of large numbers of mobile nodes will be common in the near future. Existing mobility management protocols for MANET can be classified into two broad categories [108]: routing-based protocols and location-based protocols.

**Mobility Management based on Routing**

Routing-based protocols for MANETs are typically classified into two categories: proactive routing protocols maintain up-to-date routing tables for all mobile nodes, whereas reactive routing protocols discover a route to a mobile node only when there are data packets to be delivered to the node.

Proactive routing protocols [23,24,25,26] operate in a similar way to the Internet routing protocols, as each mobile node (router) proactively maintains one or more tables for routing to other nodes in the network. To maintain a consistent view of the network topology and relative locations of mobile nodes, updates to routing tables primarily due to mobility are periodically broadcast throughout the network. Proactive routing protocols are also called table-driven routing protocols because they use and maintain routing tables. Typical examples of proactive routing protocols include the Destination-Sequenced Distance Vector (DSDV) routing protocol [23], the OLSR protocol [24], the Fisheye State Routing (FSR) protocol [25], and the Wireless Routing Protocol (WRP) [26].

Reactive routing protocols [28,29,30,31], also named source-initiated on-demand routing protocols [27], discover a route to a destination node only when the source node has data packets to be sent to the destination node. When a source node has data packets to be sent to another node and currently there is no known route to the node, the source node initiates a route discovery process, for example, by broadcasting a route discovery message throughout the network. Once the route discovery message reaches the destination node, a complete route is discovered and an acknowledgment message containing the route information is sent back to the source node along the reverse path. Examples of reactive routing protocols include the AODV routing protocol [28], the Dynamic Source Routing (DSR) protocol [29], the Associativity-Based Routing (ABR) protocol [30], and the Lightweight Mobile Routing (LMR) protocol [31].
Mobility Management based on Location Services

Due to the excessive overhead for routing information maintenance or route discovery, topology-based routing protocols generally do not scale well. Location-based routing protocols [109] offer a new direction for scalable routing in MANETs. Because they only rely on location information of neighboring nodes and the location of the destination, these protocols are highly scalable and efficient. Using a location-based routing protocol, however, the source node needs a location service [110] to first determine the location of the destination before data packets can be sent. A location service enables the storage, update, and query of location information of mobile nodes, and must be scalable to preserve the scalability of location-based routing.

Haas and Liang [32] proposed a distributed mobility management scheme for MANETs using a quorum-based location service. In this scheme, mobile nodes maintain location databases, which form a dynamic self-organizing virtual backbone within the flat network structure of a MANET. To solve the problem of intermittent wireless connectivity and node mobility that lead to unstable and inconsistent view of location databases, the scheme dynamically organizes location databases into quorums. When the location information of a mobile node needs to be updated, the location information is written to all the databases in a quorum. Similarly, when a mobile node needs to be located, its location information is read from all the databases in the quorum. Compared with traditional methods that rely on fixed location databases such as HLRs in cellular networks and HAs in Mobile IP networks, this method enhances the reliability and stability of location information of mobile nodes in a MANET, at the expense of higher signaling costs for location updates and retrievals. The scheme adapts to the dynamic mobility and network traffic patterns of mobile nodes by tuning and optimizing a parameter representing the size of quorum intersection.

Distance Routing Effect Algorithm for Mobility (DREAM) [114] is another location-based routing protocol that handles mobility management by having each node maintain a location database that stores location information about other nodes in a MANET. To keep location information stored by the nodes fresh, each node periodically broadcasts packets to update its location information with other nodes. A unique property of DREAM is that a node can control the accuracy of its location information stored by other nodes by controlling the frequency of sending the location updates (temporal resolution) and the distance a location update should travel (spatial resolution). The temporal resolution is affected by the mobility rate of the node. The spatial resolution implements the strategy that nodes closer to the source node of location updates keep more accurate
location information of that node than nodes farther away. This approach achieves a good balance
between location information accuracy and the overhead of location information maintenance.

Scalable Location Update based Routing Protocol (SLURP) [113] is a location-based routing
protocol using a location service that statically partitions the coverage area of a MANET into
equally-sized regions and assigns one of these regions as the home region for each node. The
assignment of home regions to nodes is statically calculated using a hash function known to all the
nodes. All the nodes in a node’s home region maintain location information for that node. When
a mobile node moves, it updates its location information with all the nodes in its home region by
sending location update messages. The location information of a node is defined by the region
in which the node currently resides. To locate a destination node, its home region is queried to
locate the region in which it currently resides. SLURP uses geographical forwarding named Most
Forwarding with fixed Radius (MFR) to forward packets towards a destination location.

Unlike the location service used by SLURP, Grid Location Service (GLS) [110] is a hierarchical
location service, as it divides the coverage area of a MANET into a hierarchy of square regions,
with each $n$-order region containing exactly four $n-1$-order regions. Each node stores the location
information of nodes in the same first-order region. In addition, for each node, GLS also chooses
three nodes that have the nearest node ID to store the location information of it in the three
regions at each level other than the region at the same level where it is located. Therefore, for
each node, there are some nodes at each level of the hierarchy that store location information of
it. Because the regions of higher order are larger, the density of location information of a node
decreases logarithmically with the distance from that node. This hierarchical approach maintains
a good balance between the overhead of maintaining location information and that of locating a
node.

2.1.4 Mobility Management in Wireless Mesh Networks

Existing mobility management schemes proposed for WMNs fall into three categories [7], i.e.,
tunneling-based [8,9], routing-based [6,10,11], and multicasting-based [12]. In contrast to these
existing mobility management schemes, our approach is based on pointer forwarding augmented
with location caches for efficient mobility management in WMNs.
Tunneling-based Schemes

Ant [8] is a mobility management protocol that supports intra-domain mobility within a WMN. Although the use of MAC-layer events can help Ant speedup handoff, the signaling cost of location updates in Ant is considerably high, because a location update message has to be sent to a central location server every time a mesh client changes its point of attachment. This is especially a severe problem if the mobility rate of mesh clients is high.

Huang et al. [9] proposed a mobility management for WMNs called $M^3$, which combines per-host routing and tunneling to forward packets to mesh clients. A gateway is used to host the location database and user profiles in $M^3$. $M^3$ adopts a periodic location update approach, and the location update interval is uniform for all mesh clients. In that sense, $M^3$ is not a per-user based mobility management scheme, and therefore cannot guarantee optimal performance for every mesh client.

Routing-based Schemes

iMesh [10] is an infrastructure-mode 802.11-based WMNs. iMesh is routing-based and adopts a cross-layer approach for mobility management. A link layer handoff is triggered when a mesh client moves out of the covering area of its current serving mesh router. After the link-layer handoff is completed, the routing protocol used in iMesh, the OLSR protocol, broadcasts an message announcing the new route of the mesh client. Mobility management in iMesh therefore incurs significant overhead due to the broadcasting of the message.

MEMO [11] is a WMN with support of mobility management. MEMO uses a modified AODV routing protocol, called AODV-MEMO, for integrated routing and mobility management. Like the Ant scheme, MEMO also adopts MAC-layer triggered mobility management. Although this cross-layer design (Layer 2 and 3) helps reducing the handoff latency, the use of flooding by mesh clients to inform correspondence nodes about location handoffs leads to high signaling cost and bandwidth consumption.

A common problem of iMesh and MEMO is that both of them are based on routing protocols proposed for mobile ad-hoc networks that rely on broadcasting for route discovery or location change notification, thus incurring excessive signaling overhead.

WMM [6] is a novel routing-based mobility management scheme proposed for WMNs. Location cache is used in combination with routing tables in the WMM scheme for integrated routing and location management. Because location update and location information synchronization can be
done while mesh routers route packets, the WMM scheme does not incur significant signaling overhead, compared with iMesh and MEMO. Additionally, WMM uses pointer forwarding to reduce location update cost. In our dissertation research, we use WMM as a baseline scheme against which our proposed integrated mobility and service management protocols are compared.

**Multicasting-based Schemes**

SMesh [12] offers a seamless wireless mesh network system to mesh clients, in the sense that mesh clients view the system as a single access point. Fast handoff in SMesh is achieved by using a group of mesh routers to serve a mesh client and multicast traffic to the mesh client during the handoff. This incurs a high signaling cost, which is especially a severe problem when the mobility rate of mesh clients is high. Management of multicasting groups is also a major source of signaling overhead in SMesh.

**2.2 Cache Consistency Management and Mobile Data Access**

**2.2.1 Cache Invalidation**

The issue of cache consistency management in wireless data access has been intensively studied. Most existing work focuses on cache invalidation strategies. The basic idea of cache invalidation for cache consistency management in wireless data access proposed in [58] is as follows: to ensure cache consistency, the server periodically broadcasts Invalidation Report (IR), which carry information about data objects that have been updated by the server in the most recent time interval. Clients (if active) tune to the wireless broadcast channel for IR and invalidate obsolete cached data objects according to the content of invalidation reports. To answer a query, a client needs to wait until the next IR is received to determine if there is a valid copy of the queried data object in the cache. If there is a valid copy, the query is answered immediately. Otherwise, the client sends the query to the server to retrieve a fresh copy of the data object and stores it into the cache before answering the query. A drawback of this basic approach is that the average latency for answering a query is the sum of the actual query processing time plus half of the interval between two IR broadcasts. The average latency for answering a query is large if the interval is large, whereas the signaling cost for broadcasting IRs is significant if the interval is small.

Based on this basic approach, many different IR-based cache invalidation schemes have been proposed in the literature [59,60,61,62,63,64,65,66,67]. These schemes fall into two categories: *stateful-based* versus *stateless-based*. 
Stateless-based Cache Invalidation

Stateless-based approaches are very popular and most existing schemes fall into the stateless-based category because the server does not need to keep state information [68]. In stateless-based approaches, the server either synchronously or asynchronously broadcasts IR for updated data objects. In synchronous stateless-based approaches, IR is broadcast periodically by the server, whereas in asynchronous stateless-based approaches, they are broadcast whenever data objects are updated. Asynchronous stateless-based approaches allow connected clients to keep their cache contents synchronized instantaneously. A disconnected client, however, may need to discard the entire cache contents because it has no idea about which data objects have been updated during its disconnection. Synchronous stateless-based approaches strike a balance in waiting time for both connected and disconnected clients because the waiting time for the next IR is bounded by the broadcast interval. However, synchronous stateless-based approaches have the drawback that they consume wireless bandwidth significantly for broadcasting the IR.

Cao et al. [63] proposed a scalable low-latency cache invalidation scheme that aims to reduce the average latency for answering a query as well as to efficiently utilize the broadcast bandwidth. The scheme introduces the concept of an update invalidation report (UIR). An IR contains complete history information of data object updates by the server in the past $\omega$ IR broadcast intervals. In contrast, a UIR contains only information of data object updates during an IR broadcast interval. The server divides an IR broadcast interval into $m$ sections of equal lengths and broadcasts a UIR in each section. For a connected client that receives the most recent IR, the latency for answering a query is greatly reduced because it can answer the query upon receiving the next UIR, rather than having to wait until the next IR is received. For a reconnected client that misses the most recent IR, it still has to wait until the next IR is received to validate its cache. Because a UIR does not contain complete history information of data object updates, it is generally much shorter than an IR. Therefore, broadcasting a UIR is much more cost-efficient than broadcasting an IR.

Stateful-based Cache Invalidation

In stateful-based approaches, the server is stateful as it keeps information about where data objects are cached. The CallBack (CB) approach [64,69] is a strongly consistent data access algorithm that uses the stateful-based approach to maintain strong cache consistency. In CB, whenever the server updates a data object, it asynchronously sends an IR to each client that keeps a cached copy of the updated data object. Upon receiving the IR and removing the obsolete data object, the
client sends the server an acknowledgment confirming that the invalidation is successful. If the server subsequently updates the data object before the client queries it for the first time after the invalidation, no IR needs to be sent to the client. To answer a query, the client checks if a valid copy of the queried data object exists in the cache. If there is a valid copy, the query is answered immediately. Otherwise, the client sends the query to the server to retrieve a fresh copy of the data object and stores it into the cache before answering the query. Stateful-based approaches have the same drawback as in asynchronous stateless-based approaches, i.e., a disconnected client may need to discard the entire cache contents upon reconnection because it has no idea about which data objects have been updated during its disconnection.

Xiao et al. [66] proposed an optimization to CB with two-level adaptation for wireless data access. In the first-level adaptation, the size of cache maintained by a client is adaptively adjusted, based on the so-called Update-to-Access Ratio (UAR) of data objects accessed by the client. The UAR of a data object is defined as the average number of updates to the data object between two consecutive queries for it. To determine whether a client should cache a data object it accesses, a threshold called the U-threshold is specified such that the data object should be cached only if the UAR of the data object is below the U-threshold. The U-threshold is a key parameter that affects the performance of the proposed scheme. In the second-level adaptation, when a data object is updated by the application server, the server sends the updated data object rather than an IR to those clients that keep a cached copy if the object size is smaller than the size of an IR. A push threshold denoted by T is defined such that the updated data object is sent if the object size is below the threshold.

### 2.2.2 Non-Cache-Invalidation based Approaches

Poll Each Read (PER) [64,69] is a strongly consistent data access algorithm that maintains strong cache consistency by always checking if the queried data object is still valid in the cache for every query. PER is not based on cache invalidation. Specifically, in PER, the server does not send IR when data objects are updated. Instead, to answer a query, a client always sends a message to the server to check if the cached copy of the queried data object is still valid. If the queried data object is still valid in the cache, the server sends the client an affirmative message, and the client retrieves the cached copy of the data object to answer the query. If the queried data object has been updated before the query, the server sends the updated data object to the client, and the client stores the data object into the cache before answering the query. Therefore, a cache hit in PER is not as beneficial
as in approaches based on cache invalidation. PER avoids the signaling overhead for transmitting IRs with the cost of increasing the average latency for answering a query. This algorithm incurs unnecessary signaling overhead and delay when a data object is queried more frequently than being updated by the server.

Despite that many cache consistency management schemes exist for data access in wireless environments, no existing scheme was specially designed for mobile data access in WMNs. Therefore, they cannot be used directly for cache consistency management in WMNs without considerable modification and performance penalty. For example, one important limitation of existing schemes is that they generally did not address the issues of supporting user mobility such that IR and queried data objects can be delivered properly when clients are roaming. However, supporting user mobility is required for WMNs because MC may be mobile and change their locations frequently during their stay in the networks. Another limitation of existing schemes is that clients are not able to update cached data objects, making them not appropriate for applications supporting client-side updates that are propagated to the server. In this dissertation research, we propose APPCCM to address the limitations of existing schemes. APPCCM is based on integrated cache consistency management and user mobility support as it minimizes the overall network cost incurred collectively by data access/update, cache consistency, and mobility management. APPCCM allows MC to both read and write cached data objects. Moreover, APPCCM is specifically designed for WMNs, taking into consideration of the characteristics of WMNs. For example, MR have minimal mobility and good processing power and expandable memory capacity (via USB-based flash or hard drives) [54], making it feasible to perform caching at data proxies running on the MR.

2.3 Multicast and Supporting Algorithms

2.3.1 Multicast in Mobile IP Networks
Multicast algorithms for providing mobile multicast services in Mobile IP networks have been extensively studied. Two basic approaches have been proposed in the IETF Mobile IP specification [41] for supporting mobile multicast services in Mobile IP networks, namely, Remote Subscription (RS) and Bi-directional Tunneling (BT). In RS, whenever an MN enters into a foreign network, it performs a subscription to its multicast group in the foreign network. That is, whenever the MN changes its FA, the new FA is subscribed to the multicast tree and the tree needs to be reconstructed. In this way, the optimal paths for multicast packet delivery are maintained in the presence of dynamic multicast group membership and topology due to membership changes
and user mobility. A major drawback of RS is the large signaling overhead for multicast group subscription and multicast tree reconstruction, which can be significant if mobile group members move frequently.

In BT, multicast packets are delivered to an MN in the same way in which unicast IP packets are delivered. Specifically, multicast packets are first intercepted by the HA of the MN, and subsequently delivered from the HA to the MN via standard Mobile IP tunnels. Therefore, in BT, an MN does not have to perform subscription to the multicast group when it moves and changes its location because the multicast group is only aware of the HA of each MN instead of its current location. The advantage of BT is that the signaling overhead for multicast group subscription and multicast tree reconstruction is avoided. A major disadvantage, however, is that the paths for multicast delivery in BT are generally far from optimal.

Both RS and BT have advantages and disadvantages that are complementary. Based on the idea that the advantages of the two basic approaches can be combined to offer better efficiency and scalability, various hybrid approaches [71,73,74] have been proposed to exploit the tradeoff between the cost for multicast packet delivery and the signaling cost for multicast group management and tree maintenance. mMoM is a hybrid approach that selects either RS or BT depending on the MN’s mobility characteristics. If the MN is highly mobile, BT will be used to reduce the signaling cost for multicast tree maintenance due to high mobility. Otherwise RS will be adopted to reduce the cost for multicast packet delivery. Therefore, mMoM manages to balance the tradeoff between the cost for multicast packet delivery and the signaling cost for multicast group management and tree maintenance by adaptively selecting the best approach out of RS and BT for supporting mobile multicast in Mobile IP networks.

Range-Based Mobile Multicast (RBMoM) [74] is another approach that balances the tradeoff between the service and signaling costs. RBMoM employs a special router called the Multicast Home Agent (MHA) for tunneling multicast packets to the FA to which the MN is attached. Therefore, MHA is similar to an HA except that it is exclusively used for multicast packet delivery. Like an HA in BT, each MHA is a node on the multicast tree. The HA of an MN is never changed, whereas the MHA of the MN is dynamically changed depending on the location of the MN. The HA of an MN is set to be its MHA when it initially joins the multicast group. Each MHA in RBMoM has a service range that covers a number of FAs and services MNs that are currently within its service range. The MHA of an MN is unchanged as long as the MN is roaming within the service range of its current MHA. However, when the MN moves out of the service range of its current MHA,
an MHA handoff occurs and a new MHA substitutes its current MHA to continuously provide the multicast service for the MN.

RBMoM effectively explores the tradeoff between the cost for multicast packet delivery and the signaling cost for multicast group management and tree maintenance by adjusting the service range of an MHA denoted by $R$. RS and BT are two extreme cases of RBMoM. Specifically, when $R = \infty$, RBMoM degenerates to BT because the MHA of an MN is never changed (the MHA of the MN is always its HA in this case). When $R = 0$, RBMoM is equivalent to RS because whenever an MN enters into a foreign network, the new FA becomes the new MHA of the MN and is subscribed to the multicast tree (in this case, the MHA of the MN is always the current FA to which it is attached). Therefore, RBMoM is a generalization of RS and BT.

User-oriented regional registration based mobile multicast service management (URRMoM) [71] is a regional registration based scheme for efficiently supporting mobile multicast in Mobile IP networks. URRMoM extends the design of RS with an additional layer of hierarchy consisting of Mobile Multicast Agent (MMAs) running on FAs. Each MMA provides integrated multicast service and mobility management to the MN residing in its service area. The service area of each MMA covers a number of FAs under which MNs are attached. The service area size of an MMA is therefore represented by the number of FAs it covers. Unlike in RS where the FA of the MN are multicast tree nodes, in URRMoM, the MMA forms tree nodes and are responsible for tunneling multicast packets to the FA of the MN. Multicast packets are first disseminated from the source to the MMA before they are tunneled by the MMA to the MN through the FA. URRMoM is also distinct from BT by that MNs receive multicast packets from the MMA that are dynamically selected rather than from the HA that are static.

The basic idea of regional registration based mobile multicast service management is as follows: whenever an MN moves among subnets within the service area of its MMA, it checks if its new FA already serves as an MMA. If the new FA is already an MMA and currently services other MNs, the new FA becomes the new MMA of the MN. If the new FA is not yet an MMA, it is registered with the MN’s current MMA such that the MMA can tunnel multicast packets to it. If the MN moves out of the service area of its current MMA, and if the new FA to which it moves is not an MMA, the new FA is subscribed to the multicast tree and becomes the new MMA of the MN. An MMA is unsubscribed from the multicast tree if the last MN it services leaves its service area. There exists an optimal service area size for an MMA such that the overall network cost incurred by integrated multicast service and mobility management is minimized by balancing between the
cost for multicast packet delivery and the signaling cost for multicast group management and tree maintenance. The optimal service area size is dynamically determined based on the mobility and service characteristics of the MN.

Despite that a handful of algorithms exist for supporting mobile multicast services in Mobile IP networks, these algorithms cannot be applied to WMNs directly without major modification and performance penalty. For example, those algorithms and protocols proposed for Mobile IP networks cannot be used in WMNs because WMNs lack centralized management entities such as home agents and foreign agents as in Mobile IP networks.

### 2.3.2 Multicast in Wireless Mesh Networks

The research of multicast in WMNs is still in its infancy. Very recently a few multicast algorithms and routing protocols have been proposed for WMNs [79,80,81,82,83,84,85]. Zeng et al. [79] proposed two multicast algorithms, namely, the Level Channel Assignment algorithm and the Multichannel Multicast algorithm, with the objective to improve the multicast throughput in multi-channel and multi-interface WMNs. The algorithms focus on the construction of efficient multicast trees that minimize the number of relaying nodes and the total hop count distance of the trees. By using a dedicated channel assignment strategy and partially overlapping channels, interference among channels is reduced and the throughput is improved.

Pacifier [80] is a new multicast protocol that targets high throughput and reliability. Pacifier maintains an efficient multicast tree for tree-based opportunistic multicast routing to achieve high throughput, and utilizes intra-flow network coding to achieve high reliability, without the overhead of traditional techniques such as Automatic Repeat reQuest and Forward Error Correction. Pacifier also solves the “crying baby” problem such that the throughput of well-connected nodes is improved without sacrificing the throughput of poorly-connected nodes.

In [85], two primary methods for multicast routing, namely, Shortest-Path Tree (SPT) and Minimum-Cost Tree (MCT) were investigated and evaluated via extensive simulation using a variety of performance metrics. Based on the comparative simulation results, the author recommended that SPT be used because SPT performs considerably better than MCT under identical environment conditions. In this dissertation research, we compare our proposed multicast protocol against a baseline protocol based on an SPT augmented with the capability to perform dynamic tree updates for supporting member mobility and dynamic group membership.

In [81], a cross-layer optimization framework was proposed for maximizing the multicast through-
put in WMNs. Realizing that the overall throughput tightly depends on per-link data flow rates (which further depend on link capacities controlled by radio power levels on the physical layer), the paper presented a cross-layer framework spanning the network layer, the link layer, and the physical layer. Within the framework, the multicast routing problem and the wireless medium contention problem are iteratively solved and jointly optimized to generate optimal solutions for the throughput maximization problem.

Ruiz et al. [82] proposed an integrated solution for efficient multicast routing in WMNs connected to the Internet. The solution consists two components: a tree construction algorithm that builds an approximate minimum Steiner tree for efficient multicast routing, and an auto-configuration protocol that configures MRs with topologically correct IP addresses to achieve full compatibility with standard multicast routing protocols used in the Internet.

Chakeres et al. [83] examined a wide range of multicast algorithms for WMNs based on the IEEE 802.11 standard. These algorithms provide different degrees of support to fast, efficient, and robust multicast in IEEE 802.11s WMNs. Two of these algorithms are based on broadcast, namely, Default Broadcast (DB) which is the existing multicast algorithm in IEEE 802.11s and Fast Broadcast which is an enhancement to DB. Another two algorithms are based on unicast, namely, Selective Unicast and Multiple Unicast, both of which provide robustness by L2 acknowledgments and retransmissions. The last algorithm examined is the so-called Ack-oriented in-mesh Multicast that also provides robustness by packet acknowledgment and retransmission.

While these algorithms and protocols contributed to various aspects that are key to implementing multicast in WMNs, to the best of our knowledge, the issues of supporting member mobility and dynamic group membership during the lifetime of a multicast group, which are critical in a WMN environment, have not been addressed. Specifically, existing algorithms and protocols assume static multicast trees and focus on tree construction algorithms for throughput maximization. This assumption is generally not feasible in real mobile network environments, considering that multicast group members may be highly mobile and may join or leave the group arbitrarily. Further, frequent group changes due to member mobility and dynamic group membership can cause the quality and efficiency of a static multicast tree to degrade quickly. In contrast to these algorithms and protocols, DAHM proposed in this dissertation research explicitly takes member mobility and dynamic group membership into consideration and dynamically handles mobility management and multicast service management (multicast tree maintenance, group membership management, and multicast packet delivery) in an integrated and efficient way.
2.4 Secure and Reliable Multicast and Supporting Algorithms

Algorithms proposed for secure group communications in WMNs [99,100,101] generally considered the issues related to secure group key distribution and group key agreement. In [99], a method was proposed for designing multicast key management trees that match the network topology to reduce the communication overhead associated with rekeying. A bandwidth efficient key tree management scheme was proposed in [100], focusing on assignment of key encryption keys to newly joining members to reduce the bandwidth consumption. Group key agreement in WMNs was studied in [101] by comparing three different group key agreement protocols. Pacifier [80] is a recently proposed multicast protocol for WMNs that targets high throughput and reliability. Pacifier maintains an efficient multicast tree for tree-based opportunistic multicast routing to achieve high throughput, and utilizes intra-flow network coding to achieve high reliability.

To the best of our knowledge, none of these algorithms considered user mobility support and the implication of user mobility on multicast tree maintenance, reliable multicast data delivery, security key management, and performance optimization. Unlike existing algorithms, HASRM is designed for mobile users in a WMN who may have high mobility. HASRM takes into account the effect of user mobility on secure and reliable multicast service management using an integrated design and a performance measure that combines the cost for multicast service management and the signaling cost for mobility management.

Hierarchical Reliable Multicast (HRM) [102] refers to reliable multicast algorithms that partition a multicast group into subgroups and employ a proxy in each subgroup that manages reliable multicast locally. A proxy is responsible for caching multicast data packets, collecting feedbacks from the multicast receivers, and locally retransmitting data packets in case of losses. HASRM can be considered as an instance of HRM algorithms. Unlike the work in [102], which focuses on optimizing the placement of a fixed number of static proxies on a multicast tree, HASRM dynamically determines the optimal regional service size of an MA (similar to a proxy) to minimize the network cost.
Chapter 3

System Models and Assumptions

We assume that current and future MR is powerful enough for integrated mobility and service management. Currently available wireless mesh routers already have good processing capability and expandable memory capacity (via USB-based flash or hard drives) to be used for cooperative data caching in WMN [54]. Therefore, we assume that they are also capable of performing mobility management and service management (e.g., mobile data access management, mobile multicast management, etc).

In this dissertation, we use the total communication cost incurred per time unit as the metrics for analyzing the performance of the protocols. This cost is defined by the total number of hops of wireless transmissions per time unit. The time unit is second. The total communication cost generally consists of the signaling cost for mobility management and other management operations as well as the cost for the network service of interest such as mobile data access or mobile multicast. Specific to the cost of routing a packet/message in the protocols, we assume that the shortest path is used for routing in our cost calculation. The reason is that presumably the shortest path connecting the source and destination nodes can be determined and used for routing once the location of the destination node is known. It is worth emphasizing that because the total communication cost is a per time unit measure, the accumulative effect of even a small cost difference will be significant over time.

An MC may voluntarily disconnect from the WMN and switch to idle mode periodically to reduce power consumption during its stay within the network. The MC may also involuntarily disconnect due to weak wireless connection. We view MC that is disconnected from the WMN as being in idle mode, regardless of the reasons for disconnection, as a disconnected MC cannot transmit nor receive any data, therefore incurring no network cost. We model the transition between active mode and idle mode using three parameters: $\omega$, $\omega_{w}$, and $\omega_{s}$. The physical meaning of $\omega$ is the
rate of reconnection of an MC with the WMN given that the MC is in idle mode. The reciprocal of $\omega_w$ indicates the average duration of disconnection of the MC before a transition from idle mode to active mode. Similarly, the reciprocal of $\omega_s$ denotes the average duration in which the MC keeps connected before a transition from active mode to idle mode.

To capture the mobility and service characteristics of each individual MC on a per-user basis, we use a parameter called the Service to Mobility Ratio (SMR). For an MC with an average packet arrival rate denoted by $\lambda_p$ and mobility rate denoted by $\sigma$, its SMR is defined by $\frac{\lambda_p}{\sigma}$. The physical meaning of mobility rate is the number of serving MR changes per time unit. The typical value of SMR depends on the application and the operation environment. For example, the packet arrival rate highly depends on the application. An MC that primarily uses an email application will have a much lower packet arrival rate than an MC that heavily uses video conferencing. The mobility rate also varies considerably, depending on the operation environment. In this dissertation, we focus on identifying the optimal parameter setting for minimizing the total communication dynamically, given a set of parameter values (such as SMR) characterizing the mobility and service characteristics of an MC.

For the study of per-user mobility management (Chapter 4), we assume that Internet traffic, i.e., the traffic between MR and the gateway, dominates peer-to-peer traffic in WMN [2] because WMN is expected mainly to be a solution for providing last-mile broadband Internet access. We use a parameter $\gamma$ to represent the ratio of the Internet session arrival rate to the intranet session arrival rate, and another parameter $\delta$ to represent the ratio of the average duration of Internet sessions to the average duration of intranet sessions. Internet traffic is also characterized by traffic asymmetry between the downlink and uplink [20,21]. Typically the traffic load on the downlink is much larger than the one on the uplink. Traffic asymmetry is especially pronounced for mobile multimedia applications, e.g., real-time video streaming, online radio, online games, etc. Due to traffic asymmetry, it is expected that the downlink packet arrival rate is much higher than the uplink packet arrival rate in mobile Internet applications. We use a parameter $\zeta$ to represent the ratio of the downlink packet arrival rate to the uplink packet arrival rate in Internet sessions.

For the work of routing-based mobility management with pointer forwarding (Chapter 5), we consider a WMN in which there are multiple gateways connecting the WMN to the Internet. Each gateway covers a zone of the WMN and maintains a location database for MC within the zone. For each MC, there exists an entry in the location database recording its current location information, which is the address of its forwarding chain head, i.e., the first MR on the chain. In this dissertation,
we refer to the forwarding chain head as the Anchor Mesh Router (AMR). With the address of an MC’s AMR, the MC can be located by following the forwarding chain. Note that the AMR of an MC may be co-located with its current serving MR. The zones covered by different gateways do not overlap with each other, such that at any time, the location information of any MC is kept in the location database of the gateway within which it resides.

For mobile data access (Chapter 6), we assume that MC within a WMN accesses data objects on a server that is located outside of the WMN but is accessible to the WMN through a wired connection between the server and the gateway. The data query/update characteristics between an MC and the server are specified by a number of parameters. We assume that the inter-arrival time between two consecutive queries to the same data object from the same MC follows a Poisson distribution with mean $1/\lambda$, where $\lambda$ is the query rate. Updates to the data object by the server also follow a Poisson distribution with mean $1/\mu$, where $\mu$ is the rate of updates by the server. Additionally, updates to the data object from MC follow Poisson distributions. Here we need to differentiate between two different cases: local updates from the MC under consideration and remote updates from other MC within the WMN. We use $\delta$ and $\eta$ to denote the rate of local updates by the MC under consideration and the aggregate rate of remote updates by other MC in the WMN, respectively. We define a parameter called Query to Update Ratio (QUR) to model the data query/update characteristics between the MC and the server. The QUR is given by: $QUR = \frac{\lambda}{\mu + \delta + \eta}$. Like SMR, the typical value of QUR is highly application and operation environment specific. Our research aims to show that given a value of QUR, the optimal parameter setting that minimizes the total communication cost incurred by APPCCM can always be dynamically determined.

For mobile multicast (Chapter 7), we consider a single multicast group that has a single source and dynamic group topology and membership. We assume that the multicast source is a host in the Internet. Therefore, all multicast packets must be first routed to the gateway, which then delivers them to the group members. The multicast group is dynamic with respect to both the locations of group members due to user mobility and the group membership and the group size due to member join and leave events. Within the lifetime of the multicast group, a member may join and leave the group at arbitrary time. The multicast group potentially could be highly dynamic with respect to membership during its lifetime. We assume that member join and leave events can be modeled by Poisson processes with average rates of $\lambda$ and $\mu$, respectively. We further assume that $\lambda$ and $\mu$ have equal value. This assumption ensures that the system remains in a stable state, i.e., the
steady-state multicast group size remains a constant.

For secure and reliable mobile multicast (Chapter 8), we assume that each MR possesses a pair of public/private keys and a public-key certificate that contains the identifier of the MR and its public key. There exists a central certificate authority in the WMN that is responsible for managing MR certificates. The gateway also stores the certificates and uses the public key of an MR to encrypt information (e.g., group key) to be sent to the MR. We assume that the probability of packet losses due to wireless link failure is known. HASRM achieves reliable multicast data delivery through NAK-based retransmissions [103]. Our failure model is mainly concerned with losses of multicast data packets due to mobility and unreliable wireless links. Failures of MRs are considered rare relatively to wireless link failure and are not treated. HASRM provides data secrecy via encryption to protect multicast data from unauthorized access by adversaries outside the multicast group. Specifically, HASRM aims to satisfy the forward and backward secrecy properties, i.e., it is computationally infeasible for a client to decrypt and read multicast data sent before it joins or after it leaves the multicast group. These two properties ensure that only authorized clients with a valid group membership have access to multicast data. Our security attack model considers only outside attackers. We do not consider insider attacks from entities within the multicast group.

For integrated mobility and service management in MANETs (Chapter 9), we assume that the coverage area of a MANET is statically partitioned into equally sized rectangular regions. This global partitioning of the MANET coverage area is used as the basis for home region assignment. Specifically, each mobile node is statically assigned a rectangular region whose center is also the center of the node’s home region. The home region of a mobile node is centered at one of the rectangular regions. The assignment is computed using a hash function that maps the unique ID of a mobile node (e.g., its IP or MAC address) to the ID of one of the rectangular regions. We assume that every mobile node has knowledge about the global partitioning of the MANET coverage area as well as the hash function such that it is able to locate the center of the home region of any node.

The optimal protocol setting of a protocol in terms of design parameter settings (e.g., H in Chapter 7, or $R_l$ and $R_h$ of DrMoM in Chapter 9) is optimal over perceivable ranges of parameter values. For example, $H$ in Chapter 7, which represents the threshold for the number of hops a multicast group member can be away from its multicast agent, is a positive integer and must not exceed the maximum possible distance between a member and its multicast agent.

The optimal protocol setting that minimizes the total network cost has different meanings in wired routers based mobile environments vs. wireless routers based mobile environments. In wired
routers based mobile environments, the network cost is measured by the total number of packets transmitted. In wireless routers based mobile environments especially in MANETs in which mobile routers rely on battery power, the network cost is measured by the number of wireless transmissions. This is because (1) a single wireless transmission can transmit a packet to multiple recipients within radio range of the sender; (2) sending a data packet successfully may take more than one wireless transmission due to the hidden terminal problem; and (3) wireless transmissions by wireless routers have the implication of energy consumption which is a major concern in wireless routers based mobile environments. These differences make the interpretation of the optimal protocol setting different in wired routers based mobile environments vs. wireless routers based mobile environments. In this dissertation, the optimal protocol setting is for wireless routers based mobile environments.

All protocols designed and developed in the dissertation research are dynamic, i.e., they identify and apply optimal settings to minimize network communication cost at runtime. Furthermore, they leverage cross-layer designs to obtain system parameter values characterizing an MC’s mobility and service characteristics and network conditions as input. First, an MC can measure its mobility rate when it switches from one serving MR to another. Specifically, it can monitor Signal-to-Noise Ratio (SNR) values from MRs within its radio range and the MR with the highest SNR is selected as its serving MR. This requires information regarding SNR values be supplied from the link layer to the MC for it to detect MR changes and measure its mobility rate. Second, an MC can measure its data packet rate by monitoring packet sequence numbers sent/received. This requires information be supplied from the network layer to the MC for it to calculate the data packet rate. The data packet rate information obtained when coupled with mobility rate information allows an MC to estimate its SMR. Lastly, an MC accessing data objects needs to have knowledge from the application layer regarding data query and update rates to estimate QUR as input to determine optimal settings for integrated mobility and service management at runtime.

Each protocol proposed in this dissertation depends on a number of parameters involved in the calculation of the total communication cost incurred by the protocol. We classify the parameters into three categories, namely, input parameters, derived parameters, and design parameters. Input parameters are those that measured at runtime, e.g., the mobility rate and data packet rate. Derived parameters are those that are derived from some input parameters, e.g., SMR and QUR. Design parameters are the ones that from the optimal protocol setting as discussed above, e.g., the threshold of the forwarding chain length $K$ in Chapter 4 and the local region radius $R_l$ and home
Table 3.1: Common input parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Physical meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \sigma )</td>
<td>Mobility rate</td>
</tr>
<tr>
<td>( \lambda )</td>
<td>Service rate, including data packet/session arrival rate and data object query rate</td>
</tr>
<tr>
<td>( SMR )</td>
<td>Service to mobility ratio</td>
</tr>
<tr>
<td>( \alpha )</td>
<td>Average distance (number of hops) between the gateway and an arbitrary MR</td>
</tr>
<tr>
<td>( \beta )</td>
<td>Average distance (number of hops) between two arbitrary MRs</td>
</tr>
<tr>
<td>( \tau )</td>
<td>One-hop communication latency between two neighboring MRs</td>
</tr>
<tr>
<td>( \kappa )</td>
<td>Multicast scaling factor</td>
</tr>
<tr>
<td>( N )</td>
<td>Number of MRs in the WMN</td>
</tr>
<tr>
<td>( T )</td>
<td>Multicast tree size in terms of the total number of tree nodes</td>
</tr>
</tbody>
</table>

region radius \( R_h \) in Chapter 9. Among the input parameters, some of them are protocol-specific, while the others are common to all protocols proposed in the dissertation. Here we summarize in Table 3.1 those input parameters that are common to all protocols. The physical meanings of such parameters are consistent with the ones defined in Table 3.1, unless otherwise defined. Input parameters that are protocol-specific are summarized in the chapter in which a particular protocol is discussed.
Chapter 4

Mobility Management based on Per-User Pointer Forwarding

In this chapter\(^1\), we design and analyze two mobility management schemes for WMNs based on pointer forwarding, namely, the static anchor scheme and dynamic anchor scheme. We assume that a central location database resides in the gateway. For each MC roaming in a WMN, there exists an entry in the location database that records its current location information. The location information of an MC is the address of its Anchor Mesh Router (AMR), which is the head of the MC’s forwarding chain. With the address of an MC’s AMR, the MC can be located by following the forwarding chain. Data packets sent to an MC will be routed to its current AMR first, which then forwards them to the MC by following the forwarding chain.

The forwarding chain length of an MC significantly affects the network traffic cost incurred by mobility management and packet delivery. The longer the forwarding chain, the lower rate the location update event, thus the smaller the signaling overhead. However, a long forwarding chain will increase the packet delivery cost because packets have to travel a long distance to reach the destination. Therefore, there exists a trade-off between the signaling cost incurred by mobility management vs. the service cost incurred by packet delivery. Consequently, there exists an optimal threshold of the forwarding chain length for each MC. In the proposed schemes, this optimal threshold denoted by \(K\) is determined for each individual MC dynamically, based on the MC’s specific mobility and service patterns.

We develop analytical models to evaluate the performance of the proposed schemes. We demonstrate that for both schemes, there exists an optimal threshold of the forwarding chain length that minimizes the overall network traffic incurred by mobility management and packet forwarding.

\(^1\)Part of this chapter is published as a journal paper in IEEE Transactions on Mobile Computing [70].
given a set of parameters characterizing the specific mobility and service patterns of a mesh client. We show that our schemes can yield significantly better performance than schemes that apply a static threshold to all mesh clients. Between the two proposed schemes, we show that the dynamic anchor scheme is better in typical network traffic conditions, whereas the static anchor scheme is better when the service rate of a mesh client is considerably high. We also carry out a comparative performance analysis to compare our schemes with a representative routing-based mobility management scheme.

The rest of this chapter is organized as follows. Section 4.1 and Section 4.2 introduce the static anchor scheme and dynamic anchor scheme, respectively. Performance modeling and performance analysis are carried out in Section 4.3 and 4.4, respectively. Section 4.5 summarizes this chapter.

## 4.1 Static Anchor Scheme

In the static anchor scheme, an MC’s AMR remains unchanged as long as the length of the forwarding chain does not exceed the threshold denoted by $K$.

### 4.1.1 Location Handoff

When an MC moves across the boundary of covering areas of two neighboring MRs, it de-associates from its old serving MR and re-associates with the new MR, thus incurring a location handoff. The MR it is newly associated with becomes its current serving MR. For each MC, if the length of its current forwarding chain is less than its specific threshold $K$, a new forwarding pointer will be setup between the old MR and new MR during a location handoff. On the other hand, if the length of the MC’s current forwarding chain has already reached its specific threshold $K$, a location handoff will trigger a location update. During a location update, the gateway is informed to update the location information of the MC in the location database by a location update message. The location update message is also sent to all the active Intranet correspondence nodes of the MC. After a location update, the forwarding chain is reset and the new MR becomes the AMR of the MC. Fig. 4.1 illustrates the handling of location handoffs in the proposed schemes.

### 4.1.2 Service Delivery

**Internet Session**

Internet sessions initiated towards an MC always go through the gateway, i.e., they are always routed to the gateway first before they actually enter into the WMN. Because the location database
resides in the gateway, the gateway always knows the location information of an MC by performing queries in the location database. Therefore, routing an Internet session towards an MC is straightforward. Once the location information of an MC is known, i.e., the address of the MC’s AMR is queried, the gateway can route data packets to the AMR, which then forwards them to the MC by following the forwarding chain.

**Intranet Session**

Unlike Internet sessions, which always go through the gateway where the location database is located, an Intranet session initiated towards an MC within a WMN must first determine the location information of the destination MC through a location search procedure. Suppose a mesh client MC1 initiates an Intranet session towards another mesh client MC2. Upon receiving the new session request from MC1, the serving MR of MC1 (MR1) sends a location query for MC2’s location information to the gateway, which performs the query in the location database and replies with the location information of MC2, i.e., the address of the AMR of MC2. After the location search procedure, data packets sent from MC1 to MC2 can be routed directly to the AMR of MC2, which then forwards them to MC2 by following the forwarding chain.
4.2 Dynamic Anchor Scheme

In the dynamic anchor scheme, the current forwarding chain of an MC will be reset due to the arrival of new Internet or Intranet sessions. The idea behind this scheme is to reduce the packet delivery cost by keeping the AMR of an MC close to its current serving MR when the service to mobility ratio is high, thus relieving the problem of triangular routing (gateway-AMR-MC) of the static anchor scheme, with the extra cost of resetting the forwarding chain upon a new session arrival.

The handling of location handoffs in the dynamic anchor scheme is the same as in the static anchor scheme shown in Fig. 4.1. However, the mechanism of service delivery in the dynamic anchor scheme is significantly different from that in the static anchor scheme.

4.2.1 Service Delivery

Internet Session

In the dynamic anchor scheme, when a new Internet session towards an MC arrives at the gateway, the gateway will not route the session to the AMR of the MC immediately. Instead, a location search procedure is executed to locate the MC’s current serving MR, which may be different from its AMR. Fig. 4.2 illustrates the location search procedure for newly arrived Internet sessions. Specifically, the gateway sends a location request message to the AMR of the MC, which forwards the location request to its current serving MR. Upon receiving the location request message, the MC’s current serving MR sends a location update message to the gateway, announcing that it is the new AMR of the MC. When the gateway receives the location update message, it updates the location information of the MC in the location database, i.e., marking that the current serving MR of the MC becomes its new AMR. After the location search procedure, the forwarding chain is reset and subsequent data packets will be routed to the new AMR of the MC. The gain is that the routing path is shortened, thus reducing the packet delivery cost.

Intranet Session

When a new Intranet session is initiated towards an MC, a location search procedure similar to the one above is executed to locate the current serving MR of the destination MC. Fig. 4.3 illustrates the location search procedure for newly arrived Intranet sessions. Let MC1 and MC2 denote the source mesh client and destination mesh client, respectively. When a new Intranet session initiated towards MC2 by MC1 arrives at the current serving MR of MC1 (MR1), MR1 sends a location
Figure 4.2: The location search procedure for newly arrived Internet sessions in the dynamic anchor scheme.

request message to the gateway, which queries the location database and routes the location request message to the AMR of MC2, which forwards the location request message to MC2’s current serving MR (MR2). Upon receiving the location request message, MR2 replies to the gateway with a location update message, announcing that it is the new AMR of MC2. The location information of MC2 in the location database is updated by the gateway after it receives the location reply. The updated location information of MC2 is sent to MR1 in response to the location request and the location search procedure is completed. After the location search procedure, subsequent data packets will be routed to the new AMR of MC2 directly.

4.3 Performance Modeling

In this section, we develop analytical models for evaluating the performance of the proposed schemes. The analytical models are built using stochastic Petri nets (SPN). Table 4.1 summarizes the parameters and notations used in the following sections.
Table 4.1: The parameters and notations used in performance modeling and analysis.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Physical meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\sigma$</td>
<td>Input</td>
<td>Mobility rate</td>
</tr>
<tr>
<td>$\lambda_I/\mu_I$</td>
<td>Input</td>
<td>Internet session arrival/departure rate</td>
</tr>
<tr>
<td>$\lambda_L/\mu_L$</td>
<td>Derived</td>
<td>Intranet session arrival/departure rate</td>
</tr>
<tr>
<td>$\lambda_{pu}$</td>
<td>Derived</td>
<td>Average uplink (outcoming) packet arrival rate of Internet sessions</td>
</tr>
<tr>
<td>$\lambda_{pd}$</td>
<td>Derived</td>
<td>Average downlink packet arrival rate of Internet sessions</td>
</tr>
<tr>
<td>$\lambda_{pL}$</td>
<td>Derived</td>
<td>Average packet arrival rate of Intranet sessions</td>
</tr>
<tr>
<td>$N_{I}$</td>
<td>Input</td>
<td>Average number of downlink (incoming) packets per Internet session</td>
</tr>
<tr>
<td>$N_{pL}$</td>
<td>Input</td>
<td>Average number of incoming packets per Intranet session</td>
</tr>
<tr>
<td>$N_{I}$</td>
<td>Derived</td>
<td>Instantaneous average number of active Internet correspondence nodes per MC</td>
</tr>
<tr>
<td>$N_{pL}$</td>
<td>Derived</td>
<td>Instantaneous average number of active Intranet correspondence nodes per MC</td>
</tr>
<tr>
<td>$N$</td>
<td>Input</td>
<td>Number of MRs in a WMN</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>Input</td>
<td>Average distance (number of hops) between the gateway and an arbitrary MR</td>
</tr>
<tr>
<td>$\beta$</td>
<td>Input</td>
<td>Average distance (number of hops) between two arbitrary MRs</td>
</tr>
<tr>
<td>$\gamma$</td>
<td>Input</td>
<td>Ratio of the Internet session arrival rate to the Intranet session arrival rate</td>
</tr>
<tr>
<td>$\delta$</td>
<td>Input</td>
<td>Ratio of the average duration of Internet sessions to the one of Intranet sessions</td>
</tr>
<tr>
<td>$\zeta$</td>
<td>Input</td>
<td>Ratio of the downlink packet arrival rate to the uplink packet arrival rate of Internet sessions</td>
</tr>
<tr>
<td>$\tau$</td>
<td>Input</td>
<td>One-hop communication latency between two neighboring MRs</td>
</tr>
<tr>
<td>$\omega$</td>
<td>Derived</td>
<td>Rate of reconnection when an MC switches from sleep mode back to active mode</td>
</tr>
<tr>
<td>$P_f$</td>
<td>Derived</td>
<td>Probability that an MC moves forward</td>
</tr>
<tr>
<td>$P_b$</td>
<td>Derived</td>
<td>Probability that an MC moves backward</td>
</tr>
<tr>
<td>$P_g$</td>
<td>Input</td>
<td>Probability that an Intranet packet is routed to the gateway due to unknown location information of the destination MC in the WMM scheme</td>
</tr>
<tr>
<td>$P_q$</td>
<td>Input</td>
<td>Probability that the location query procedure is executed in the WMM scheme</td>
</tr>
<tr>
<td>$P_r$</td>
<td>Input</td>
<td>Probability that an MR broadcasts the route request message in the WMM scheme</td>
</tr>
</tbody>
</table>
4.3.1 Static Anchor Scheme

The SPN model for the static anchor scheme is shown in Fig. 4.4. This model essentially captures the behaviors of an MC while it is moving around within a WMN. The interpretation of places and transitions defined in the SPN model is given in Table 4.2. In Fig. 4.4 we put in numbers in parenthesis to label the SPN model sequence below. Here we briefly describe how the SPN model is constructed:
Table 4.2: The interpretation of places and transitions defined in the SPN model for the static anchor scheme.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Move</td>
<td>A timed transition modeling MC movement</td>
</tr>
<tr>
<td>Movement</td>
<td>( \text{Mark(Movement)}=1 ) means that the MC just moved</td>
</tr>
<tr>
<td>Forward</td>
<td>An immediate transition modeling forward movement</td>
</tr>
<tr>
<td>Backward</td>
<td>An immediate transition modeling backward movement</td>
</tr>
<tr>
<td>NewMR</td>
<td>( \text{Mark(NewMR)}=1 ) means that the MC just moved forward to a new MR</td>
</tr>
<tr>
<td>PreMR</td>
<td>( \text{Mark(PreMR)}=1 ) means that the MC just moved backward to the most recently visited MR</td>
</tr>
<tr>
<td>AddPointer</td>
<td>A timed transition modeling the event of setting up a forwarding pointer between two neighboring MRs</td>
</tr>
<tr>
<td>FL</td>
<td>( \text{Mark(FL)} ) indicates the forwarding chain length</td>
</tr>
<tr>
<td>ResetLU</td>
<td>A timed transition modeling the event of updating the location database and resetting the forwarding chain</td>
</tr>
<tr>
<td>RemPointer</td>
<td>An immediate transition modeling the event of removing a forwarding pointer due to a backward movement</td>
</tr>
</tbody>
</table>

- The movement of an MC is modeled by transition \textit{Move}, the transition rate of which is represented by the mobility rate \( \sigma \) of an MC. When an MC moves to a new MR and is re-associated with it, thus incurring a location handoff, a new token is put into place \textit{Movement}, meaning that the location handoff is completed.

- An MC can move forward to a new MR, or move backward to the most recently visited MR. The SPN model differentiates between these two cases using two immediate transitions, i.e., \textit{Forward} and \textit{Backward}. Probability \( P_f \) and \( P_b \) are associated with \textit{Forward} and \textit{Backward}, respectively. The values of \( P_f \) and \( P_b \) depend on the network coverage model used, which will be introduced in Section 4.3.3.

- If an MC moves forward to a new MR, transition \textit{Forward} is fired and a new token is put into place \textit{NewMR}. If the current forwarding chain length is smaller than \( K \), a new forwarding pointer needs to be setup. This is modeled by enabling and firing transition \textit{AddPointer}, if the number of tokens in place \textit{FL} is less than \( K \).

- If the number of tokens in place \textit{FL} is already equal to \( K \), a new forward movement triggers a location update and the forwarding chain is reset. This is modeled by firing transition \textit{ResetLU}, when there are \( K \) tokens in place \textit{FL} and a token in place \textit{NewMR}. The firing of transition \textit{ResetLU} will consume all the \( K \) tokens in place \textit{FL}, representing that the
forwarding chain is reset.

- If an MC moves backward to the most recently visited MR, transition *Backward* is fired and a token is put into place *PreMR*. This will subsequently enable and fire immediate transition *RemPointer*. Removing a pointer upon a backward movement is modeled by an immediate transition as forwarding pointers will be purged automatically.

- Notice that it is only reasonable for an MC to move backward, when the current serving MR of the MC is not its AMR, i.e., the forwarding chain length is not zero. This is modeled by associating an enabling function (*Mark(FL) > 0*) with transition *Backward*.

- The inhibitor arcs in the SPN model are used to model the assumption that an MC will not move during a location handoff.

- The transition rates of transition *AddPointer* and *ResetLU* are parameterized in Section 4.3.3.

### 4.3.2 Dynamic Anchor Scheme

The SPN model for the dynamic anchor scheme is shown in Fig. 4.5. Because the handling of location handoffs is the same in both schemes, part of this SPN model is identical to the SPN model for the static anchor scheme. The SPN model for the dynamic anchor scheme has 4 new transitions and 2 new places to capture the behavior of the scheme when new sessions arrive toward
Table 4.3: The interpretation of additional places and transitions defined in the SPN model for the dynamic anchor scheme.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISessionArrival</td>
<td>A timed transition modeling the arrival of Internet sessions</td>
</tr>
<tr>
<td>LSessionArrival</td>
<td>A timed transition modeling the arrival of Intranet sessions</td>
</tr>
<tr>
<td>ISession</td>
<td>A place holding newly arrived Internet sessions</td>
</tr>
<tr>
<td>LSession</td>
<td>A place holding newly arrived Intranet sessions</td>
</tr>
<tr>
<td>ResetIS</td>
<td>A timed transition modeling resetting the forwarding chain due to a newly arrived Internet session</td>
</tr>
<tr>
<td>ResetLS</td>
<td>A timed transition modeling resetting the forwarding chain due to a newly arrived Intranet session</td>
</tr>
</tbody>
</table>

The event of new (Internet or Intranet) session arrival towards an MC is modeled by firing transition \( ISessionArrival \) or \( LSessionArrival \), the transition rates of which are \( \lambda_I \) and \( \lambda_L \), respectively. The firing of transition \( ISessionArrival \) or \( LSessionArrival \) causes a token to be put into place \( ISession \) or \( LSession \), depending on the type of the new session.

In the dynamic anchor scheme, the arrival of a new session causes the current forwarding chain to be reset, and the new MR to become its new AMR. This is modeled by firing transition \( ResetIS \) for a newly arrived Internet session or \( ResetLS \) for a newly arrived Intranet session. The firing of transition \( ResetIS \) or \( ResetLS \) consumes all the tokens in place \( FL \), modeling that the current forwarding chain is reset upon a new session arrival.

The inhibitor arcs are used to model the assumption that new sessions will not arrive during the course of a forwarding chain reset.

The transition rates of transition \( ResetIS \) and \( ResetLS \) are parameterized in Section 4.3.3.

4.3.3 Parameterization

Transition \( AddPointer \) models the event of setting up a forwarding pointer between two neighboring MRs, which involves a round-trip communication between the two MRs, i.e., the communication cost is \( 2\tau \). The transition rate \( \mu_{AddPointer} \) is the reciprocal of the communication delay, i.e.:
\[ \mu_{AddPointer} = \frac{1}{2\tau} \] (4.1)

Transition ResetLU models the event of resetting the forwarding chain of an MC during a location update, which involves updating the MC’s location information in the location database, and sending a location update message to each active Intranet correspondence node (CN) of the MC. The signaling cost thus consists of two parts. The first part is for the new MR to inform the gateway to update the MC’s location information in the location database, i.e., \( \alpha \tau \). The second part is for the new MR to inform all the active Intranet CNs of the MC, i.e., \( N_L \beta \tau \). Thus, the transition rate \( \mu_{ResetLU} \) is:

\[ \mu_{ResetLU} = \frac{1}{(\alpha + N_L \beta) \times \tau} \] (4.2)

Transition ResetIS models the event of resetting the forwarding chain of an MC due to the arrival of a new Internet session. As introduced in Section 4.2.1, in this event, a location request message is sent from the gateway to the current serving MR of the MC, and a location update message is replied to the gateway in response to the location request. The location update message is also sent to all the active Intranet CNs of the MC. The communication cost in this case is therefore \((2\alpha + N_L \beta + i) \times \tau\), where \( i \) is the current length of the forwarding chain. Thus, the transition rate \( \mu_{ResetIS} \) is:

\[ \mu_{ResetIS} = \frac{1}{(2\alpha + N_L \beta + i) \times \tau} \] (4.3)

Transition ResetLS models the event of resetting the forwarding chain of an MC due to the arrival of a new Intranet session. Let MC1 and MC2 denote the source MC and destination MC, respectively. In this event, a location request message is sent from the serving MR (MR1) of MC1 to the current serving MR (MR2) of MC2, forwarded by the gateway. In response to the location request, a location update message is replied by MR2 to the gateway, which then forwards the updated location information of MC2 to MR1. The location update message is also sent to all the active Intranet CNs of MC2. The communication cost in this case is thus \((4\alpha + N_L \beta + i) \times \tau\), where \( i \) is the current length of the forwarding chain. Thus, the transition rate \( \mu_{ResetLS} \) is:

\[ \mu_{ResetLS} = \frac{1}{(4\alpha + N_L \beta + i) \times \tau} \] (4.4)
Immediate transitions Forward and Backward are associated with probability $P_f$ and $P_b$, respectively. These probabilities depend on the network coverage model and the mobility model assumed. We assume the square-grid mesh network model for WMNs [94] and the random walk model [14] for MC. For the square-grid mesh network model, we assume that all MRs have the same wireless range that covers direct neighboring MRs located in four orthogonal directions. Additionally, we consider a relatively large wireless mesh network simulated by a wrapped-around structure such that each MR has four direct neighbors. Under these models, an MC can move randomly from the current MR to one of the MR’s four neighbors with equal probability, i.e., 1/4. Thus, $P_f$ and $P_b$ can be calculated as:

$$P_f = \frac{3}{4}, P_b = \frac{1}{4}$$ (4.5)

Packet arrival rates, e.g., $\lambda_{pu}$, $\lambda_{pd}$, and $\lambda_{pL}$, are effective rates over the continuous time space. Because packets arrive only when there are on-going sessions, these rates depend on session arrival rates, e.g., $\lambda_I$ and $\lambda_L$, and average numbers of packets per session, e.g., $N_{pI}$ and $N_{pL}$. We use a $M/M/\infty$ queue to model the process of session arrival towards an MC. The average number of on-going sessions of an MC at any instance can be obtained using queuing theory, and thus the effective packet arrival rate can be derived. Specifically, the average number of on-going Internet (Intranet) sessions of an MC denoted by $N_I$ ($N_L$) at any instance is calculated as:

$$N_I = \lambda_I \frac{\mu_I}{\mu_L}, \quad N_L = \lambda_L \frac{\mu_L}{\mu_I}, \text{with } \lambda_L = \frac{\lambda_I}{\gamma} \text{ and } \mu_L = \delta \times \mu_I$$ (4.6)

Notice that in Equation 4.6, we state that $\mu_L = \delta \times \mu_I$. This is because according to queuing theory, the ratio of the average duration of Internet sessions to the average duration of Intranet denoted by $\delta$ is defined as: $\delta = \frac{1/\mu_I}{1/\mu_L} = \frac{\mu_L}{\mu_I}$.

The effective downlink (incoming) and uplink (outcoming) packet arrival rates of Internet sessions and the effective packet arrival rate of Intranet sessions are derived as:

$$\lambda_{pd} = N_{pI} \times N_I \times \lambda_I$$

$$\lambda_{pu} = \frac{\lambda_{pd}}{\zeta}$$

$$\lambda_{pL} = N_{pL} \times N_L \times \lambda_L$$ (4.7)

4.3.4 Performance Metrics

We use the total communication cost incurred per time unit as the metrics for performance evaluation and analysis. The total communication cost includes the signaling cost of location handoff and
update operations, the signaling cost of location search operations, and the packet delivery cost. For the static anchor scheme, the signaling cost of location search operations is incurred when a new Intranet session is initiated towards an MC. For the dynamic anchor scheme, the signaling cost of location search operations represents the cost for tracking the current serving MR of an MC and resetting the forwarding chain when new sessions are initiated towards an MC. In the following, we use $C_{\text{static}}$ and $C_{\text{dynamic}}$ to represent the total communication cost incurred per time unit by the static anchor scheme and dynamic anchor scheme, respectively. $C_{\text{location}}$, $C_{\text{search}}$, and $C_{\text{delivery}}$ are used to represent the signaling cost of a location handoff operation, the signaling cost of a location search operation, and the cost to deliver a packet, respectively. Subscripts are associated with these cost terms. Specifically, subscript “T” and “L” denote Internet and Intranet sessions, respectively. Subscript “s” and “d” denote the static anchor scheme and dynamic anchor scheme, respectively.

For the static anchor scheme, the total communication cost incurred per time unit is calculated as:

$$C_{\text{static}} = C_{\text{location}} \times \sigma + C_{\text{search,L}} \times \lambda_L + C_{\text{delivery,I}} \times \lambda_{pd} + C_{\text{delivery,L}} \times \lambda_{pL} \quad (4.8)$$

For the dynamic anchor scheme, the total communication cost incurred per time unit is calculated:

$$C_{\text{dynamic}} = C_{\text{location}} \times \sigma + C_{\text{search,I}} \times \lambda_I + C_{\text{search,L}} \times \lambda_L + C_{\text{delivery,I}} \times \lambda_{pd} + C_{\text{delivery,L}} \times \lambda_{pL} \quad (4.9)$$

The stochastic models underlying the SPN models shown in Fig. 4.4 and Fig. 4.5 are continuous-time Markov chains. Let $P_i$ denote the probability that the underlying Markov chain is found in a state that the current forwarding chain length is $i$. Let $S$ denote the set of states in the underlying Markov chain. Then $C_{\text{location}}$ can be calculated as:

$$C_{\text{location}} = \sum_S P_i C_{i,\text{location}} \quad (4.10)$$

where $C_{i,\text{location}}$ is calculated as:

$$C_{i,\text{location}} = \begin{cases} 2\tau & \text{if } 1 \leq i < K \\ (\alpha + N_L \beta) \times \tau & \text{if } i = K \end{cases} \quad (4.11)$$

The location search cost $C_{\text{search}}$ can be calculated as:
\[ C_{\text{search}} = \sum_{S} P_{i} C_{i,\text{search}} \]  
\( (4.12) \)

where \( C_{i,\text{search}} \) is either \( C_{i,\text{search},s,L} \), \( C_{i,\text{search},d,I} \), or \( C_{i,\text{search},d,L} \). The equations for calculating \( C_{i,\text{search},s,L} \), \( C_{i,\text{search},d,I} \), and \( C_{i,\text{search},d,L} \) are shown as follows:

\[ C_{i,\text{search},s,L} = 2\alpha \tau \]
\[ C_{i,\text{search},d,I} = (2\alpha + N_L\beta + i) \times \tau \]
\[ C_{i,\text{search},d,L} = (4\alpha + N_L\beta + i) \times \tau \]  
\( (4.13) \)

The packet delivery cost \( C_{\text{delivery}} \) is calculated in a similar way as follows:

\[ C_{\text{delivery}} = \sum_{S} P_{i} C_{i,\text{delivery}} \]  
\( (4.14) \)

where \( C_{i,\text{delivery}} \) is either \( C_{i,\text{delivery},I} \) or \( C_{i,\text{delivery},L} \). \( C_{i,\text{delivery},I} \) and \( C_{i,\text{delivery},L} \) can be calculated as follows:

\[ C_{i,\text{delivery},I} = (\alpha + i) \times \tau \]
\[ C_{i,\text{delivery},L} = (\beta + i) \times \tau \]  
\( (4.15) \)

These costs can be calculated by associating the above SPN models with reward functions and calculating the steady-state rewards, using the SPNP [15] package.

### 4.4 Performance Analysis and Numerical Results

In this section, we analyze the performance of the proposed schemes, in terms of the total communication cost incurred per time unit. Additionally, we compare the proposed schemes with two baseline schemes. In the first baseline scheme, pointer forwarding is not used, meaning that every movement of an MC will trigger a location update event. Thus it is essentially the same as having \( K = 0 \) in the proposed schemes. This baseline scheme is chosen in order to demonstrate the benefit of pointer forwarding. In the second baseline scheme, pointer forwarding is employed, but the same threshold of the forwarding chain length is preset for all MC, e.g., \( K = 4 \) for all MC. This baseline scheme is selected to justify the benefit of dynamically determining the optimal threshold of the forwarding chain length on a per-user basis. We also carry out performance comparison between our schemes and the WMM scheme proposed in [6]. A detailed description of the WMM scheme and the SPN model constructed for it will be given in Section 4.4.3. Table 4.4 lists the parameters and their default values used in the performance evaluation. The time unit used is second. All costs presented below are normalized with respect to \( \tau = 1 \).
Table 4.4: The parameters and their default values used.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Parameter</th>
<th>Value</th>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\gamma$</td>
<td>10</td>
<td>$\delta$</td>
<td>5</td>
<td>$\lambda_I$</td>
<td>$\frac{1}{500}$</td>
</tr>
<tr>
<td>$\mu_I$</td>
<td>$\frac{1}{1000}$</td>
<td>$N_I$</td>
<td>200</td>
<td>$N_L$</td>
<td>100</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>30</td>
<td>$\beta$</td>
<td>30</td>
<td>$N$</td>
<td>1000</td>
</tr>
<tr>
<td>$P_g$</td>
<td>5.0%</td>
<td>$P_q$</td>
<td>10.0%</td>
<td>$P_r$</td>
<td>50.0%</td>
</tr>
<tr>
<td>$\tau$</td>
<td>1</td>
<td>$\omega_{active}$</td>
<td>$\frac{1}{1200}$</td>
<td>$\omega_{sleep}$</td>
<td>$\frac{1}{600}$</td>
</tr>
</tbody>
</table>

4.4.1 Proposed Pointer Forwarding Schemes

Fig. 4.6: Total communication cost vs. $K$.

Fig. 4.6 shows the total communication cost as a function of $K$ in both schemes, under different SMRs. As shown in the figure, for both schemes, there exists an optimal threshold $K$ that results in minimized total communication cost. For example, when $SMR = 1$, the optimal $K$ is 10 for the static anchor scheme, whereas it is 11 for the dynamic anchor scheme. Another observation is that the total communication cost in both schemes decreases, as SMR increases. This is because given fixed session arrival rates, the mobility rate decreases as SMR increases, thus the signaling cost incurred by location management as well as the total communication cost decreases.

It is interesting to note in Fig. 4.6 that the dynamic anchor scheme always performs better than the static anchor scheme, under the given parameter values in Table 4.4 and the investigated SMRs. However, since the dynamic anchor scheme incurs additional overhead of resetting the forwarding chain of an MC upon session arrival, it is expected that in cases that session arrival rates are considerably high, the additional overhead will offset its advantage. This is demonstrated by Fig. 4.7, which plots the cost difference between the static anchor scheme and dynamic anchor.
scheme, as a function of SMR, with $\lambda_I = \frac{1}{30}$ and $\mu_I = \frac{1}{30}$.

![Figure 4.7: $C_{\text{static}} - C_{\text{dynamic}}$ vs. SMR.](image)

It can be seen in Fig. 4.7 that initially when SMR is small, the dynamic anchor scheme performs better than the static anchor scheme. However, as SMR increases, there exists a crossover point beyond which the static anchor scheme starts performing better than the dynamic anchor scheme. It is interesting to see that there exists another crossover point of SMR beyond which the dynamic anchor scheme is superior again. This is because when SMR is considerably large, i.e., when the mobility rate is considerably small relatively to the session arrival rate, resetting the forwarding chain due to new session arrival in the dynamic anchor scheme essentially makes the AMR of an MC be the same as its current serving MR most of the time, thus significantly reducing the packet delivery cost. It is worth noting that because the total communication cost is a per time unit measure, the accumulative effect of even a small cost difference will be significant.

![Figure 4.8: Optimal $K$ vs. SMR.](image)
Fig. 4.8 plots the optimal threshold $K$ as a function of SMR in both schemes. It can be observed that for both schemes, the optimal $K$ decreases, as SMR increases. This is because as SMR increases, with fixed session arrival rates, the mobility rate decreases, thus a short forwarding chain is favorable to reduce the service delivery cost. It is also interesting to see that the optimal $K$ in the static anchor scheme is always smaller than or equal to the one in the dynamic anchor scheme, due to resetting the forwarding chain of an MC upon new session arrival in the dynamic anchor scheme.

### 4.4.2 Proposed Schemes vs. Baseline Schemes

![Cost difference vs. SMR between the proposed schemes and baseline schemes.](image)

Figure 4.9: Cost difference vs. SMR between the proposed schemes and baseline schemes.

Fig. 4.9 shows the difference of the total communication cost between the proposed schemes and baseline schemes, as a function of SMR. For example, $C_{K=4} - C_{\text{dynamic}}$ in the figure means the difference of the total communication cost between the dynamic anchor scheme and the baseline scheme in which a fixed threshold $K = 4$ is applied to all MC. The minimum total communication cost under the optimal threshold $K$ is used for the proposed schemes. It can be seen in the figure that the proposed schemes perform significantly better than both baseline schemes, especially when SMR is small. The cost differences decrease as SMR increases. The reason is that given fixed session arrival rates, the mobility rate and accordingly the signaling cost incurred by location management decrease as SMR increases, in both the proposed schemes and baseline schemes. As can be seen in the figure, however, the proposed schemes are always superior to both baseline schemes. The comparison demonstrates the advantage of identifying the optimal threshold of the forwarding chain length on a per-user basis.
4.4.3 The WMM Scheme

WMM [6] is a routing-based mobility management scheme, in the sense that location management is integrated with packet routing. This idea is earlier adopted by Cellular IP [16] and HAWAII [17], both of which are routing-based micro-mobility management schemes for Mobile IP networks. Each MR in WMM employs a proxy table to store the location information of MC for which it has routed packets. Location information is also carried out in the IP header of every data packet. Location information is synchronized in every MR along the route of a packet between the MR’s proxy table and the IP header of the packet, based on the relative magnitude of time stamps they carry.

In order to route Internet packets to an MC, the gateway must know the location information of the MC, i.e., the address of the MC’s current serving MR. If the MC’s location information is not found in the gateway’s proxy table, a location query procedure based on broadcasting must be executed, which incurs significant overhead. If the MC’s location information stored in the gateway’s proxy table is fresh, packets can be routed to the destination directly. On the other hand, if the stored location information is obsolete, packets will be routed to the obsolete serving MR first, which then forwards them to the MC’s current serving MR. This is similar to delivering packets by following a forwarding chain.

To route an Intranet packet, the current serving MR (MR1) of the source MC (MC1) uses different routing strategies, depending on whether its local proxy table has an entry for the destination MC (MC2). If its proxy table has an entry for MC2, MR1 routes the packet to the recorded serving MR of MC2; otherwise, it routes the packet to the gateway. In the first case, if the recorded serving MR of MC2 is obsolete, the packet will be routed to the obsolete serving MR first, which then forwards the packet to MC2’s current serving MR. This is again similar to the forwarding chain approach.

Based on the above discussion, we argue that the WMM scheme can be viewed as a variant of mobility management schemes based on pointer forwarding. The first Internet packet originated from an MC after its most recent location handoff essentially serves as a location update message in the WMM scheme. Additionally, an Intranet packet originated from an MC that is routed to the gateway due to unknown location information of the destination is essentially a location update message as well. Between two consecutive (Internet or Intranet) packets originated from an MC arriving at the gateway, movement of the MC may trigger a series of location handoffs and a chain of proxy table entries is setup along the path of its movement. Such a chain of proxy table entries
is similar to a chain of forwarding pointers. In this sense, each data packet originated from an MC arriving at the gateway essentially resets the forwarding chain, because the gateway’s proxy table is updated according to the location information of the MC carried in the packet.

The average time interval between two reset operations is the same as the average inter-arrival time between two data packets originated from an MC reaching the gateway. Let $P_g$ denote the probability that an Intranet packet originated from an MC arrives at the gateway due to unknown location information of the destination. The effective arrival rate of packets originated from an MC reaching the gateway is therefore $\lambda_{pu} + P_g \times \lambda_{pL}$, and the average inter-arrival time $T_{lu}$ of such two consecutive packets can be calculated as:

$$T_{lu} = \frac{1}{\lambda_{pu} + P_g \times \lambda_{pL}}$$

(4.16)

The average distance (number of hops) an MC can move during the time interval $T_{lu}$ is therefore $M = T_{lu} \times \sigma$, which is essentially the threshold of distance an MC can move between two consecutive forwarding chain reset operations. It is important to realize that generally the distance of movement is different from the forwarding chain length, because backward movement reduces the forwarding chain length by 1.

![Figure 4.10: The SPN model for the WMM scheme.](image)

Based on the above observation, we develop an SPN model for the WMM scheme, as shown in Fig. 4.10. Notice that we use separate places $NM$ and $FL$ to represent the distance of movement and the forwarding chain length, respectively. As discussed above, a threshold $M$ is associated with the arc from place $NM$ to transition $LocUpdate$. Due to the space limit, we omit the description of how the SPN model is constructed.

The total communication cost incurred by the WMM scheme, denoted by $C_{wmm}$, consists of
the signaling cost incurred by handling location handoffs, the packet delivery cost, and the location query cost.

\[ C_{wmm} = C_{\text{location}} \times \sigma + C_{\text{delivery}, I} \times \lambda_{pd} + C_{\text{delivery}, L} \times \lambda_{pL} + C_{\text{query}} \times P_q \times \omega \]  

(4.17)

Location handoffs are handled in the WMM scheme by a registration procedure, which involves a round-trip communication between two neighboring MRs. Therefore \( C_{\text{location}} = 2\tau \) in the WMM scheme. Let \( i \) denote the number of tokens in place \( FL \) in Fig. 4.10, i.e., \( i \) is the length of the forwarding chain, then the cost of Internet packet delivery is calculated as: \( C_{\text{delivery}, I} = (\alpha + i) \times \tau \).

Intranet sessions in WMNs, which involve two peers interacting with each other bi-directionally, usually have similar packet arrival rates in both directions. It indicates that the location information of each peer stored by the serving MR of the other peer is updated in a similar rate. Thus, packets sent and received between two peers involved in an Intranet session usually travel the same distance \( \beta \) on the average. The delivery cost of Intranet packets denoted by \( C_{\text{delivery}, L} \) is therefore \( \beta \tau \) in the WMM scheme.

The location query procedure, executed by the gateway, is required only when there are packets to be sent to an MC before the MC initiates the first Internet session, after 1) the MC newly enters into a WMN; or 2) the MC wakes up and reconnects to the WMN after staying in sleep mode for some time (an MC may voluntarily disconnect from a WMN and switch to sleep mode to save battery life). In both cases, packets to be sent to the MC will be routed to the gateway, because the MC’s current serving MR is unknown. The probability denoted by \( P_q \) that the location query procedure is executed in the above cases is investigated in [6]. Typically, an MC switches alternatively between active mode and sleep mode during its stay in a WMN. Let \( \omega_w \) and \( \omega_s \) denote the rate of switching from sleep mode to active mode and the rate of reverse mode switching, respectively. Let \( \omega \) in Equation 4.17 denote the reconnection rate of an MC. Then \( \omega \) can be calculated as:

\[ \omega = \frac{1}{\omega_w} + \frac{1}{\omega_s} = \frac{\omega_w \times \omega_s}{\omega_w + \omega_s} \]  

(4.18)

To locate an MC whose current serving MR is unknown, the location query procedure is executed by broadcasting a route request message to all MRs. The current serving MR of the MC replies to the gateway a route response message upon receiving the route request message. Thus, the signaling cost of the location query procedure denoted by \( C_{\text{query}} \) is the sum of the cost of broadcasting
the route request message and the cost of transmitting the route response message. The cost of transmitting the route response message is $\alpha \tau$. Here we present a brief analysis of the cost of broadcasting the route request message. In the following, we define the cost of broadcasting a route request message as the number of broadcasts required to deliver the message to all MRs, instead of the sum of one-hop transmission costs, because of the broadcasting nature of wireless transmission. It is important to realize that this definition of the broadcasting cost is appropriate only by assuming that all MRs use the same omnidirectional wireless channel for communication. For future multi-radio multi-channel WMNs that are built upon advanced radio techniques, e.g., cognitive radio and directional antenna, the cost would be underestimated.

We assume that a flooding algorithm based on self-pruning [18] is used for broadcasting in WMNs. Self-pruning utilizes the knowledge of direct neighborhood of each node to reduce redundant rebroadcasts, which is a serious problem in flooding-based broadcasting algorithms, commonly referred to as the broadcast storm problem [19]. In self-pruning, each node $\nu$ maintains a list of direct neighbors, denoted by $N(\nu)$. A node $\nu_j$ who receives a flooding packet from its neighbor $\nu_i$ rebroadcasts the packet only if $N(\nu_i) - N(\nu_j)$ is nonempty and it is the first time $\nu_j$ receives the packet. This indicates that each node will rebroadcast a flooding packet no more than once. Thus, the cost of flooding the route request message in WMNs can be calculated as: $P_r \times N$, where $P_r$ denotes the average probability that an MR rebroadcasts the route request message, and $N$ denotes the number of MRs. Therefore, we have $C_{query} = P_r \times N + \alpha \tau$.

### 4.4.4 Proposed Schemes vs. The WMM Scheme

To analyze the performance of the WMM scheme, we introduce the parameter $\zeta$, which represents the ratio of the downlink packet arrival rate to the uplink packet arrival rate of Internet sessions, based on the observation of traffic asymmetry between the downlink and uplink [20,21,22], i.e., typically the traffic load on the downlink is much larger than the one on the uplink. Traffic asymmetry is especially pronounced for mobile multimedia applications, e.g., real-time audio/video streaming, online radio, online interactive games, etc., because in such applications, small content requests are transmitted via the uplink, whereas the requested content that is typically large is transmitted via the downlink. Due to traffic asymmetry, it is expected that the downlink packet arrival rate is much higher than the uplink packet arrival rate in mobile Internet applications, i.e., $\zeta$ is expected to be reasonably large.

Fig. 4.11 plots the total communication cost per time unit incurred by the WMM scheme
denoted by $C_{wmm}$ as a function of $\zeta$, under different SMRs. As can be seen in the figure, $C_{wmm}$ increases almost linearly as $\zeta$ increases. This is because as $\zeta$ increases, $M$ and accordingly the forwarding chain length increase, thus causing the packet delivery cost to increase. As in our schemes, $C_{wmm}$ also increases as SMR increases. It is also interesting to see in the figure that the slope of the cost curve is inversely proportional to SMR, i.e., it is in direct proportion to the mobility rate. Therefore, when the mobility rate is high, the performance of the WMM scheme degrades quickly as $\zeta$ increases. This is a major drawback of routing-based mobility management schemes, in that the propagation of updated location information of MC relies on packet routing, thereby incurring possibly significant delay.

Fig. 4.11 also shows the performance of the dynamic anchor scheme under the investigated SMRs as references. The performance data of the dynamic anchor scheme shown in Fig. 4.11 takes into consideration of the cost incurred when an MC newly enters into a WMN and each time when the MC wakes up and reconnects to the WMN. Let $C_{reconnect}$ denote the cost. In the proposed schemes, a location update message is sent to the gateway when either one of these two events happens. Thus, $C_{reconnect} = \alpha \tau$ in the proposed schemes. It can be seen in Fig. 4.11 that under a specific SMR, there exists a crossover point of $\zeta$ beyond which the dynamic anchor scheme performs better than the WMM scheme. Let $\zeta_c$ denote the crossover point.

Fig. 4.12 plots $\zeta_c$ as a function of SMR, under different query probabilities. It is interesting to observe that each curve in Fig. 4.12 exhibits a shape consisting of two segments with clearly different trends. This is the result of varying contributions to the total communication cost as SMR increases between the signaling cost incurred by location management and the service delivery cost. Specifically, when SMR is small, i.e., when the mobility rate is high, the contribution of
the signaling cost incurred by location management is significant, and it is much larger in the
dynamic anchor scheme than in the WMM scheme, because location update in the WMM scheme
incurs minimum overhead. Therefore, when SMR is small, a relatively large $\zeta_c$ is necessary for
the dynamic anchor scheme to yield better performance than the WMM scheme, and $\zeta_c$ roughly
decreases as SMR increases. As SMR increases, the contribution of the signaling cost incurred
by location management becomes less significant, whereas the relative contribution of the service
delivery cost increases. Intuitively, there exists a point of SMR beyond which the trend is shifted,
i.e., $\zeta_c$ starts increasing as SMR increases. As indicated above, the WMM scheme favors small
mobility rates, because its performance drops quickly when the mobility rate is high. As SMR
increases, the mobility rate decreases, therefore the minimum $\zeta_c$ required for the dynamic anchor
scheme to perform better than the WMM scheme, i.e., $\zeta_c$, increases.

Fig. 4.12 also shows that the query probability denoted by $P_q$ significantly affects the perfor-
mance of the WMM scheme. As investigated in [6], $P_q$ is dependent on various parameters, and
can range from lower than 5% to higher than 95%. We expect that $P_q$ will not be significantly low
in normal network traffic conditions. For example, unless $\gamma$ is extremely large, e.g., $\gamma \geq 1000$, $P_q$
will typically be higher than 10%, and may even be above 50%, according to the analysis of $P_q$
presented in [6]. Therefore, we expect that the typical range of $\zeta_c$ is [10, 100] in normal network traffic
conditions. As discussed above, $\zeta$ is expected to be large for mobile Internet applications, due to
traffic asymmetry between the downlink and uplink. Thus we argue that this range is reasonable,
and we conclude that the dynamic anchor scheme is superior to the WMM scheme, when mobile
Internet applications dominate the network traffic in WMNs.
4.4.5 Sensitivity Analysis

In this section, we investigate the sensitivity of analytical results with respect to various parameters characterizing the network condition and structure, e.g., \( \alpha \), \( \beta \), and the network coverage model assumed.

As can be seen in Section 4.3.4, \( \alpha \) and \( \beta \) are two critical parameters that determine the cost of mobility management as well as the cost of packet delivery. Fig. 4.13 compares the total communication cost between the proposed schemes and the baseline schemes, under two different combinations of \( \alpha \) and \( \beta \): 1) \( \alpha = 20, \beta = 20 \), and 2) \( \alpha = 30, \beta = 30 \). As expected, under the same SMR, the total communication cost increases as \( \alpha \) and \( \beta \) increase. However, regardless of the values of \( \alpha \) and \( \beta \), the trend remains the same, i.e., the proposed schemes perform significantly better than the baseline schemes, especially when SMR is small. This observation conforms to the result illustrated in Fig. 4.9.

As introduced in Section 4.3.3, we assume the square-grid mesh network model for WMNs and the random walk model for MC. In order to investigate the effect of network coverage models on the performance of the proposed schemes, we switch to the hexagonal network coverage model as used in [6]. In the hexagonal network coverage model, the coverage area of each MR is called a cell, and each MR has six direct neighbors. An MC can move randomly from an MR to one of its direct neighbors with the same probability. Thus, \( P_f = \frac{5}{6} \) and \( P_b = \frac{1}{6} \), under the hexagonal network coverage model.

Fig. 4.14 plots the total communication cost incurred by the proposed schemes as a function of \( K \), under different SMRs, assuming the hexagonal network coverage model. Comparing Fig. 4.6
and Fig. 4.14, it can be seen that cost curves shown in both figures exhibit high similarity in shape. Therefore, we can draw the conclusion that analytical results obtained are valid and are not sensitive to the network coverage model.

4.5 Summary

In this chapter, we proposed two mobility management schemes based on pointer forwarding for wireless mesh networks, namely, the static anchor scheme and dynamic anchor scheme. The proposed schemes are per-user based, in that the optimal threshold of the forwarding chain length that minimizes the total communication cost is dynamically determined for each individual MC, based on the MC’s specific mobility and service patterns characterized by SMR.

We developed analytical models based on stochastic Petri nets to evaluate the proposed schemes. We also compared the proposed schemes with two baseline schemes and with the WMM scheme. Analytical results showed that (1) the dynamic anchor scheme is better than the static anchor scheme in typical network traffic conditions, whereas the static anchor scheme is better when the service rate of an MC is comparatively high such that the advantage of the dynamic anchor scheme is offset by the extra cost; (2) our schemes significantly outperform the baseline schemes, especially when SMR is small; (3) the dynamic anchor scheme is superior to the WMM scheme when network traffic is dominated by mobile Internet applications characterized by large traffic asymmetry for which the downlink packet arrival rate is much higher than the uplink packet arrival rate.
Chapter 5

Routing-based Mobility Management with Pointer Forwarding

In this chapter\(^1\), we design and analyze LMMesh, a routing-based mobility management scheme with pointer forwarding for WMNs. LMMesh integrates routing-based location update and pointer forwarding into a single scheme that exploits the advantages of both methods, while avoiding their drawbacks. A well-known advantage of routing-based location update is that it enables the propagation of location information of MC to the concerning parties using regular data packets originated from the MC, therefore avoiding the signaling overhead of explicit location update messages. Routing-based location update, however, does not work well for MC that do not have active network sessions or MC that are not sending data packets. Pointer forwarding is a solution for location management that uses explicit location update messages. It works for those MC for which routing based location update does not work well, at the expense of additional signaling cost for the location update messages.

Although routing-based location update and pointer forwarding have been individually applied in many mobile communication networking studies, the integration of them and the impact of this integration on the overall network performance has not been studied. The contributions of this work are (a) we formulate the interaction between routing-based location update and pointer forwarding and analyze the impact of this integration on the overall network communication cost incurred; (b) we propose the design notion of optimal pointer forwarding when it is integrated with routing-based location update by dynamically identifying the optimal pointer forwarding chain length for each MC based on the MC’s service and mobility characteristics to minimize the network communication cost; (c) we develop an analytical model based on stochastic Petri net [15] techniques for performance

\(^1\)Part of this chapter is published as a journal paper in IEEE Transactions on Network and Service Management [118].
analysis; through a comparative performance study, we show that LMMesh outperforms both pure routing-based location management schemes and pure pointer forwarding schemes, as well as traditional tunnel-based location management schemes.

The rest of this chapter is organized as follows. Section 5.1 presents the proposed location management scheme. In Section 5.2 we develop a performance model to analyze the performance of LMMesh. Section 5.3 presents numerical data to demonstrate the effectiveness of LMMesh and to compare its performance against existing location management schemes for WMNs. The chapter is summarized in Section 5.4.

5.1 LMMesh

In this section, we present the proposed location management scheme, namely LMMesh. In Sections 5.1.1-5.1.4 we discuss the protocol behavior when a MC is within a gateway zone. In Section 5.1.5, we discuss the protocol behavior when a MC moves from one gateway zone to another. Finally in Section 5.1.6, we address the scalability of LMMesh.

5.1.1 Routing-based Location Update and Pointer Forwarding

In LMMesh, we allow every data packet (in an Internet or intranet session) originated from an MC to carry the up-to-date location information of the sender, i.e., the address of the MC’s current serving MR, in the option field of the packet header. Upon receiving the data packet, a gateway (in an Internet session) or an intranet correspondence node (CN) of the MC (in an intranet session) extracts the location information from the data packet. The gateway uses this information to update the location database, and the intranet CN uses this information to route data packets to the MC. More specifically, for an Internet session between an MC and an Internet host, when receiving an uplink data packet from the MC, the gateway uses the location information carried by the data packet to update the location database. For an intranet session between two MCs in the same WMN, location information carried by data packets transmitted between the MCs is used by the serving MR of the receiver to update its routing table and to route data packets to the sender.

Routing-based location update works well for MCs that are actively sending data packets. For MCs that do not have active network sessions or MCs that are not sending data packets, however, routing-based location update does not work well. Even for MCs that are actively sending data packets, routing-based location update may not be a complete solution. For example, suppose that there is an intranet session between \( MC_1 \) and \( MC_2 \) in the same WMN, and that \( MC_1 \) continuously
sends data packets to \( MC_2 \). Although \( MC_2 \) is continuously being updated with the up-to-date location information of \( MC_1 \), the gateway may not be updated because data packets from \( MC_1 \) to \( MC_2 \) may not go through the gateway. Now, suppose that an Internet host initiates a new Internet session towards \( MC_1 \). Upon receiving the session, the gateway may need to perform a costly location query procedure based on broadcasting, as in [6], to locate \( MC_1 \) before delivering the session.

To address those problems, LMMesh uses a per-user pointer forwarding method to complement routing-based location update. The basic pointer forwarding method [13] works as follows. When an \( MC \) moves from its current serving \( MR \) to a new \( MR \), a location handoff is performed. If the length of the \( MC \)'s current forwarding chain is less than a threshold \( K \) for the forwarding chain length, a new forwarding pointer is setup between the old \( MR \) and new \( MR \), and the forwarding chain length is increased by one. The physical representation of a forwarding pointer kept by an \( MR \) is the address of the next \( MR \) along the forwarding chain. On the other hand, if the length of the forwarding chain is equal to \( K \), no new forwarding pointer can be setup in this case. Instead, the movement triggers a location update, i.e., a location update message is sent to the gateway to update the \( MC \)'s location information stored in the location database. After the location update, the forwarding chain is reset and the new serving \( MR \) becomes the new AMR of the \( MC \). Note
that the forwarding chain is also reset whenever the gateway receives a data packet from the MC that carries the address of the MC’s current serving MR. Fig. 5.1 illustrates the pointer forwarding method using an example in which $K = 2$, described below:

1. When the MC moves from its current AMR, which is also its current serving MR, to MR1, a forwarding pointer is setup between its AMR and MR1 and the forwarding chain length is one;

2. The MC moves to MR2 after employing MR1 as its serving MR for some time, and a forwarding pointer is setup between MR1 and MR2 and the forwarding chain length becomes two;

3. The MC again moves, this time to MR3, after being associated with MR2 for some time. This third movement causes the forwarding chain being reset because $K = 2$;

4. MR3 becomes the MC’s new AMR, and a location update message is sent to the gateway to update the MC’s location information stored in the location database.

LMMesh takes a step further by dynamically determining the optimal threshold for the forwarding chain length that minimizes the overall communication cost for each individual MC, based on the MC’s specific mobility and service characteristics. The overall communication cost includes the signaling cost for location management and the service cost for packet delivery. The forwarding chain length of an MC significantly affects the overall communication cost incurred by the MC. Specifically, the longer the forwarding chain, the lower the location update rate, and consequently, the smaller the signaling overhead. However, the packet delivery cost increases as the forwarding chain becomes longer. Intuitively, there exists a trade-off between the signaling cost for location management and the service cost for packet delivery. LMMesh explores the tradeoff and dynamically determines the optimal threshold for the forwarding chain length that minimizes the overall communication cost on a per-user basis. In the remainder of the chapter, we use $K$ to denote the threshold, and $K_{optimal}$ to represent the optimal threshold. We show that the analytical model developed in this chapter can be used to dynamically determine $K_{optimal}$, given parameters characterizing the specific mobility and service characteristics of an MC.
5.1.2 Integration and Its Impact

LMMesh uses both methods for location management in an integrated manner that achieves network cost minimization dynamically on a per-user basis. The integration takes the advantages of both methods, while avoiding their drawbacks. The use of routing-based location update has a positive effect of reducing the signaling traffic of explicit location update messages in pointer forwarding. The reason is that LMMesh relies less on the explicit location update messages in pointer forwarding for location management when an MC is actively sending data packets. On the other hand, when an MC does not have active network sessions or is not sending data packets, the use of pointer forwarding addresses the problems associated with routing-based location update, as discussed above. Particularly, the costly location query procedure based on broadcasting as in [6] is avoided by using the pointer forwarding method.

Essentially, LMMesh is adaptive to the changing mobility and service behaviors of an MC in the context of the integration. This adaption is the result of dynamically determining the optimal threshold $K_{\text{optimal}}$ for the forwarding chain length. The value of $K_{\text{optimal}}$ changes dynamically when the MC has service and mobility activities that vary over time. For example, when the rate at which the MC sends data packets is high, the value of $K_{\text{optimal}}$ of the MC tends to increase. Consequently, the rate at which location update messages are sent in pointer forwarding tends to decrease. This is because LMMesh relies less on the explicit location update messages for location management when an MC is actively sending data packets that also serve the purpose of implicit location update messages. These observations are demonstrated by the numerical results presented in Section 5.3.

5.1.3 Location Search Procedure

When a new Internet or intranet session is initiated towards an MC, LMMesh utilizes a location search procedure to locate the current serving MR of the MC before the session is delivered. By locating the current serving MR of the MC and consequently resetting the forwarding chain, the new session can be delivered directly to the MC following the shortest path, thereby reducing the packet delivery cost. The gain in the reduction of the packet delivery cost is particularly pronounced when the packet arrival rate to the MC is considerably high, compared with its mobility rate. Note that the location search procedure is only executed when a new session is initiated towards an MC.
Location Search for Internet Sessions

Fig. 5.2 illustrates the location search procedure for a new Internet session initiated towards an MC, which is described as follows:

1. When an Internet session initiated by an Internet host towards an MC arrives at the gateway, the gateway sends a location request message to the MC’s current AMR (the gateway keeps the address of the MC’s AMR in the location database);
2. The AMR forwards the message to the MC’s current serving MR;
3. Upon receiving the location request message, the MC’s current serving MR sends a location update message back to the gateway, making itself the new AMR of the MC;
4. The gateway updates the location information of the MC in the location database, and the location search procedure is completed.

After the location search procedure is completed, the MC’s forwarding chain is reset and subsequent downlink Internet data packets from the Internet host to the MC will be routed to the new AMR. The gain is that the routing path is shortened, and the packet delivery cost is reduced.

Location Search for Intranet Sessions

For an intranet session initiated by $MC_1$ towards $MC_2$, a location search procedure similar to the above is executed to locate the current serving MR (MR2) of MC2. Fig. 5.3 illustrates the procedure, described below:

1. When the current serving MR (MR1) of MC1 receives the intranet session request from MC1, it sends a location request message to the gateway;
2. The gateway forwards the message to MC2’s current AMR;
3. The message is further forwarded to MR2 following the forwarding chain;
4. Upon receiving the location request message, MR2 sends a location update message back to the gateway;
5. The gateway updates the location information of MC2 in the location database;
6. The gateway also sends the updated location information of MC2 (the address of MC2’s new AMR) to MR1, and the location search procedure is completed.

In this procedure, after the gateway receives the location update message, the current forwarding chain of MC2 is reset and MR2 becomes its new AMR. Subsequent data packets sent to MC2 will be routed to its new AMR.

5.1.4 Data Packet Routing

To route data packets to an MC in LMMesh, the address of the MC’s current AMR must be known. This information is always kept in the location database on the gateway. It is also carried in the packet header of data packets originated from the MC (the address carried in this case is the one of the MC’s current serving MR). Once the address of the MC’s current AMR is known, routing data packets to the MC simply relies on the underlying routing protocol.

Specifically, in an Internet session, data packets sent from an Internet host to an MC always pass through the gateway, which routes them to the current AMR of the MC as recorded in the location database. The AMR forwards the data packets to the MC’s current serving MR (if the AMR and current serving MR are not co-located), following the forwarding chain, and the serving MR finally delivers the packets to the MC. In an intranet session between two MCs, data packets sent between the MCs are first routed from the sender MC’s current serving MR to the receiver MC’s current AMR, which then forwards the packets to the receiver MC’s current serving MR (if the receiver MC’s AMR and serving MR are not co-located). The serving MR finally delivers the
data packets to the receiver MC.

5.1.5 Multiple Gateways

When an MC moves from one zone to another, a gateway-level location handoff occurs to transfer
the mobility management role from the gateway of the MC’s current zone to that of its new zone.

Specifically, when the MC moves to a new zone, it first registers with the gateway of the
new zone and obtain a new gateway foreign address (GFA), by sending a location binding update
message to the gateway. When the gateway receives the message, it creates a new entry for the
MC with the address of the MC’s new AMR, which is the new serving MR of the MC in the new
zone. The MC also sends a location binding update message to all its current intranet and Internet
CNs such that future network traffic from these CNs to the MC will be routed towards the new
gateway. Before the CNs are updated with the new GFA of the MC, however, they will send data
packets to the MC’s old gateway. To prevent those data packets from being lost, the MC sends its
old gateway a location binding cancellation message carrying its new GFA. When the old gateway
receives the message, it knows that the MC is registered with the new gateway and forwards data
packets received from the MC’s CNs towards the new gateway for the time period before the CNs
are updated with the MC’s new GFA. After the gateway-level location handoff, the MC executes
LMMesh as described in Sections 5.1.1-5.1.4 for mobility management within the new zone.

When a gateway receives a request for the current location of an MC for which it cannot find
an entry in its location database, the gateway broadcasts the request to all the other gateways.
Upon receiving a reply from another gateway that has the current location information of the MC,
the gateway sends the current location information of the MC, i.e., the address of the MC’s current
AMR to the requester. It is worth emphasizing that LMMesh can minimize the probability of such
costly broadcasting traffic through the gateway-level location handoff procedure and the integration
of routing-based location update and pointer forwarding.

5.1.6 Scalability of LMMesh

Scalability is an important requirement for a mobility management scheme for WMNs. LMMesh is a
scalable solution for mobility management because it supports sharing of the mobility management
role at both the gateway and MR levels in a hierarchical way such that a single gateway or MR
will not become the bottleneck.

- Each gateway is only responsible for mobility management of MCs within the zone. The
mobility management responsibility of a gateway for an MC is transferred to another gateway once the MC migrates to another zone. Therefore, given that gateways are typically placed in the WMN with load balancing principles (e.g., [93]), the gateways will evenly share the mobility management responsibility. No gateway would become the bottleneck.

- Within each zone, all MRs share the mobility management load by talking the role of either an AMR or a forwarding MR on the forwarding chain of an MC. Furthermore, because AMRs are dynamically selected by the MCs when they move, no MR would become the bottleneck by taking the role of an AMR.

- The benefit of minimizing the network communication cost on a per MC basis as a result of applying LMMesh is cumulative and proportional to the number of MCs. This design consideration makes LMMesh especially beneficial for large WMNs.

5.2 Performance Model

5.2.1 Analytical Model for LMMesh

In this section, we develop an analytical model based on stochastic Petri net (SPN) techniques for analyzing the performance of LMMesh. Table 5.1 lists the parameters and their physical meanings used in the following sections. Fig. 5.4 shows the SPN model for LMMesh. The SPN model captures the dynamic service and mobility behavior of an MC using states and events. We choose SPN as the tool for performance modeling because: 1) an SPN model is a concise representation of the underlying Markov or semi-Markov chain that may have a large number of states; 2) an SPN model is capable of reasoning the behavior of an MC, as it migrates among states in response to system events.

An SPN model consists of entities such as transitions (e.g., Move and Forward), tokens, places (e.g., Movement and FL), and arcs that connect transitions and places. A transition is used to represent the firing of an event, and it can be either a timed transition (e.g., Move and ResetLU) or an immediate transition (e.g., Forward and Backward). A timed transition is fired after an event occurrence time is elapsed, while an immediately transition fires immediately. For example, an MC moves to a new MR (modeled by firing transition Move) after being associated with its current serving MR for an amount of time that is exponentially distributed. A token is used as a marker; it is used here to represent an event occurrence. For example, a new token is put into place Movement when Move is fired. A place is a token holder to contain tokens which represent the
Table 5.1: Parameters and their physical meanings used in performance modeling and analysis.

<table>
<thead>
<tr>
<th>Parameter Type</th>
<th>Physical meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\sigma$</td>
<td>Input Mobility rate</td>
</tr>
<tr>
<td>$\lambda_{UI}/\lambda_{DI}$</td>
<td>Input Uplink/downlink Internet session arrival rate</td>
</tr>
<tr>
<td>$\lambda_{IL}/\lambda_{OL}$</td>
<td>Derived Incoming/outgoing intranet session arrival rate</td>
</tr>
<tr>
<td>$\mu_{I}/\mu_{L}$</td>
<td>Input/Derived Internet/intranet session departure rate</td>
</tr>
<tr>
<td>$\lambda_{UIP}/\lambda_{DIP}$</td>
<td>Derived Uplink/downlink packet arrival rate of Internet sessions</td>
</tr>
<tr>
<td>$\lambda_{ILP}/\lambda_{OLP}$</td>
<td>Input Incoming/outgoing packet arrival rate of intranet sessions</td>
</tr>
<tr>
<td>$\omega$</td>
<td>Derived Rate of reconnections to the WMN</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>Input Average number of hops between the gateway and an MR</td>
</tr>
<tr>
<td>$\beta$</td>
<td>Input Average number of hops between any two MRs</td>
</tr>
<tr>
<td>$\gamma$</td>
<td>Input Ratio of the Internet session arrival rate to the intranet session arrival rate</td>
</tr>
<tr>
<td>$\delta$</td>
<td>Input Ratio of the average duration of Internet sessions to that of intranet sessions</td>
</tr>
<tr>
<td>$\zeta$</td>
<td>Input $\frac{\lambda_{UIP}}{\lambda_{DIP}}$</td>
</tr>
<tr>
<td>$\tau$</td>
<td>Input One-hop communication latency between two neighboring MRs</td>
</tr>
<tr>
<td>$P_f/P_b$</td>
<td>Derived Probability that an MC moves forward/backward</td>
</tr>
<tr>
<td>$P_g$</td>
<td>Input Probability that an intranet packet is routed by the gateway</td>
</tr>
<tr>
<td>$P_q$</td>
<td>Input Probability that the location query procedure is executed in WMM</td>
</tr>
<tr>
<td>$P_r$</td>
<td>Input Probability that an MR broadcasts the route request message in WMM</td>
</tr>
<tr>
<td>$N_{MR}$</td>
<td>Input Number of MRs in a WMN</td>
</tr>
</tbody>
</table>

number of event occurrences. For example, the number of tokens in place $FL$ is used to represent the forwarding chain length. Finally, an output arc connects a transition to a place and an input arc connects a place to a transition. An arc is associated with a multiplicity defining the number of tokens that will be moved into the output place (if it is an output arc) or moved out of the input place (if it is an input arc). For example, the arc that connects place $FL$ to transition $ResetLU$ has a multiplicity of $K$. This means that when transition $ResetLU$ fires, it consumes $K$ tokens from place $FL$.

In Fig. 5.4 we put in numbers in parenthesis to label the SPN model sequence below. The SPN model for LMMesh is constructed as follows:

1. The movement of an MC is modeled by transition $Move$, the transition rate of which is $\sigma$.
   When the MC moves to a new MR, thus incurring a location handoff, a new token is put into place $Movement$, indicating that the location handoff is completed.

2. The MC may move forward to a new MR, or move backward to the most recently visited MR. The SPN model differentiates between these two cases using two immediate transitions
**Forward and Backward.** Probabilities $P_f$ and $P_b$ associated with **Forward** and **Backward** depend on the network coverage model, which will be introduced in Section 5.2.2.

3. If the MC moves forward to a new MR, transition **Forward** is fired and a new token is put into place **NewMR**. If the current forwarding chain length is smaller than $K$, a new forwarding pointer needs to be setup. This is modeled by firing transition **AddPointer**, if the number of tokens in place **FL** is less than $K$. **FL** represents the current forwarding chain length.

4. If the number of tokens in place **FL** is already equal to $K$, a new forward movement triggers a location update and the forwarding chain is reset. This is modeled by firing transition **ResetLU**, when there are $K$ tokens in place **FL** and one token in place **NewMR**. The firing of **ResetLU** consumes all tokens in place **FL**, representing that the forwarding chain is reset.

5. If the MC moves backward to the most recently visited MR, transition **Backward** is fired and a token is put into place **PreMR**. This will subsequently enable and fire immediate transition
RemPointer. We use an immediate transition to model the event of removing a forwarding pointer because the pointer will be purged automatically upon timeout.

6. Notice that it is only reasonable for the MC to move backward, when the forwarding chain length is not zero. This is modeled by associating an enabling function \((\#(FL) > 0)\) with transition Backward.

7. The arrival of a new uplink Internet session initiated by the MC is modeled by firing transition UISArrival, the transition rate of which is \(\lambda_{UI}\). Accordingly, the arrival of a new downlink Internet session towards the MC is modeled by transition DISArrival, the transition rate of which is \(\lambda_{DI}\).

8. The arrival of a new outgoing intranet session initiated by the MC is modeled by firing transition OLSArrival, the transition rate of which is \(\lambda_{OL}\). Accordingly, the arrival of a new incoming intranet session towards the MC is modeled by transition ILSArrival, the transition rate of which is \(\lambda_{IL}\).

9. Transitions ISDeparture and LSDeparture are used to model Internet session departure and intranet session departure, respectively.

10. The arrival of a new session towards the MC triggers the location search procedure and causes its current forwarding chain to be reset. This is modeled by firing transition ResetDIS for a newly arrived downlink Internet session or ResetILS for a newly arrived incoming intranet session. In either case, all tokens in place FL are consumed, modeling that its current forwarding chain is reset.

11. The location information of the MC stored in the location database is also updated when the gateway receives from the MC an uplink Internet data packet or an intranet data packet that is to be forwarded by the gateway. The arrival of these two kinds of data packets is modeled by transitions UIPArrival and OLPArrival, respectively. The events of updating the location database in these two cases are modeled by firing immediate transitions ResetUIP and ResetOLP. The firing of ResetUIP or ResetOLP consumes all tokens in place FL, representing that the current forwarding chain is reset.

12. It is only possible for the MC to send data packets when it has on-going Internet or intranet sessions. This is modeled by associating enabling functions as shown in the SPN model with
transitions *UIPArrival* and *OLPArrival*.

13. The MC switches alternatively between active mode and sleep (idle) mode. Initially the MC is in active mode, and can send and receive data packets. After staying in active mode for a period of time, the MC switches to sleep mode to save battery life. This is modeled by firing transition *Active2Sleep* and putting a token into place *Sleep*. The transition rate of *Active2Sleep* is $\omega_s$. When the MC is in sleep mode, it will not incur any network communication activities. The MC wakes up after being in sleep mode for some time. This is modeled by firing transition *Sleep2Active* and putting a token into place *Active*. The transition rate of *Sleep2Active* is $\omega_w$.

14. When the MC wakes up and reconnects to the WMN, it sends a location binding update message to the gateway. This event is modeled by firing transition *LocUpdate*.

15. We assume that the MC switches to sleep mode only when it has no on-going sessions. This is modeled by associating an enabling function with transition *Active2Sleep*. When the MC is in sleep mode, it will not have any network activities and will not incur any location handoff. This condition is modeled by associating enabling functions with transitions *DISArrival*, *UISArrival*, *OLSArrival*, *ILSArrival*, *UIPArrival*, *OLPArrival*, and *Move*.

### 5.2.2 Parameterization

Transition *AddPointer* models the event of setting up a forwarding pointer. In this case, a round-trip message exchange between two involving MRs is carried out, thus the communication cost is $2\tau$. The transition rate is the reciprocal of the communication delay, i.e.,

$$\mu_{AddPointer} = \frac{1}{2\tau} \quad (5.1)$$

Transition *ResetLU* models the event of resetting the forwarding chain of an MC during a location update. This involves a round-trip message exchange between the gateway and the MC’s current serving MR. The signaling cost incurred is $2\alpha\tau$. Thus, the transition rate is:

$$\mu_{ResetLU} = \frac{1}{2\alpha\tau} \quad (5.2)$$

Transition *ResetIS* models the event of resetting the forwarding chain of an MC due to the arrival of a new Internet session. Let $i$ denote the length of the current forwarding chain of an
MC. As elaborated in Section 5.1.3, the communication cost in this case is \((2\alpha + i) \times \tau\). Thus, the transition rate is:

\[
\mu_{\text{ResetIS}} = \frac{1}{(2\alpha + i) \times \tau}
\]  

(5.3)

Transition \(\text{ResetLS}\) models the event of resetting the forwarding chain of an MC due to the arrival of a new intranet session. As elaborated in Section 5.1.3, the communication cost in this case is \((4\alpha + i) \times \tau\). Thus, the transition rate is:

\[
\mu_{\text{ResetLS}} = \frac{1}{(4\alpha + i) \times \tau}
\]  

(5.4)

Transition \(\text{LocUpdate}\) models the event of sending the gateway a location binding update message when an MC wakes up and reconnects. The gateway replies with a location binding confirmation message as an acknowledgment. The signaling cost incurred is \(2\alpha\tau\). Thus, the transition rate is:

\[
\mu_{\text{LocUpdate}} = \frac{1}{2\alpha\tau}
\]  

(5.5)

Transition \(\text{UIPArrival}\) models the arrival of uplink Internet data packets originated from an MC at the gateway. The transition rate of \(\text{UIPArrival}\) is the effective rate of uplink Internet data packets originated from the MC, which can be calculated as follows:

\[
\mu_{\text{UIPArrival}} = \text{mark}(\text{ISessions}) \times \lambda_{\text{UIP}}
\]  

(5.6)

where \(\text{mark}(\text{ISessions})\) returns the number of tokens in place \(\text{ISessions}\), i.e., the number of ongoing Internet sessions of the MC.

The arrival of outgoing intranet data packets originated from an MC at the gateway is modeled by transition \(\text{OLPArrival}\). The transition rate of \(\text{OLPArrival}\) is the effective rate of outgoing intranet data packets originated from the MC arriving at the gateway, which can be calculated as:

\[
\mu_{\text{OLPArrival}} = P_g \times \text{mark}(\text{LSessions}) \times \lambda_{\text{OLP}}
\]  

(5.7)

where \(P_g\) is as defined in Table 5.1, and \(\text{mark}(\text{LSessions})\) represents the number of ongoing intranet sessions.

The transition rates of \(\text{ISDeparture}\) and \(\text{LSDeparture}\) are effective departure rates of Internet and intranet sessions, respectively. We use a \(M/M/\infty\) queue to model the process of session arrivals towards an MC. Using the \(M/M/\infty\) queuing model, the transition rates of \(\text{ISDeparture}\) and \(\text{LSDeparture}\) can be derived as follows:

\[
\mu_{\text{ISDeparture}} = \text{mark}(\text{ISessions}) \times \mu_I
\]
\[
\mu_{\text{LSDeparture}} = \text{mark}(\text{LSessions}) \times \mu_L
\]  

(5.8)
The outgoing and incoming intranet session arrival rates and the uplink Internet packet arrival rate can be calculated as follows:

\[
\begin{align*}
\lambda_{OL} &= \frac{\lambda_{UI}}{\gamma} \\
\lambda_{IL} &= \frac{\lambda_{DI}}{\gamma} \\
\lambda_{UIP} &= \frac{\lambda_{DIP}}{\zeta}
\end{align*}
\]  

We assume the square-grid mesh network model for WMNs [94] and the random walk model for MCs. For the square-grid mesh network model, we assume that all MRs have the same wireless range that covers direct neighboring MRs located in four orthogonal directions. Under these models, each MR has four direct neighbors and an MC can move randomly from the current MR to one of the MR’s neighbors with equal probability, i.e., \(1/4\). Thus, we have:

\[P_f = \frac{3}{4}, P_b = \frac{1}{4}\]  

An MC typically switches alternatively between active mode and sleep mode during its stay in a WMN. The rate of reconnection denoted by \(\omega\) can be derived as follows:

\[\omega = \frac{\omega_w \times \omega_s}{\omega_w + \omega_s}\]  

5.2.3 Performance Metrics

We use the total communication cost incurred per time unit as the metric for performance evaluation and analysis. It is worth noting that because the total communication cost is a per time unit measure, the accumulative effect of even a small cost difference will be significant.

The total communication cost incurred per time unit by LMMesh consists of the signaling cost of location handoff and update operations, the signaling cost of location tracking operations, the signaling cost of location binding update upon reconnect, and the packet delivery cost. Let \(C_{LMMesh}\) denote the total communication cost incurred per time unit by LMMesh, and let \(C_{location}, C_{tracking}, C_{reconnection},\) and \(C_{delivery}\) denote the cost components, respectively. Subscripts “I” and “L” denote Internet and intranet sessions, respectively. Using these cost terms, \(C_{LMMesh}\) is calculated as follows:

\[
C_{LMMesh} = C_{location} \times \sigma' + C_{tracking,I} \times \lambda'_{DI} + C_{tracking,L} \times \lambda'_{IL} + C_{delivery,I} \times \lambda'_{DIP} + C_{delivery,L} \times \lambda'_{ILP} + C_{reconnection} \times \omega
\]  

In the above equation, \(\sigma'\) represents the steady-state effective mobility rate. \(\lambda'_{DI}\) and \(\lambda'_{IL}\) represent the steady-state effective downlink Internet session arrival rate and incoming intranet session arrival rate, respectively. \(\lambda'_{DIP}\) and \(\lambda'_{ILP}\) denote the steady-state aggregate downlink Internet
packet arrival rate and incoming intranet packet arrival rate, respectively. The first three rates are “effective” rates to account for the fact that when an MC is in sleep mode it will not incur network communication activities, and they can be calculated by:

\[
\begin{align*}
\sigma' &= (1 - P_{\text{Sleep}}) \times \sigma \\
\lambda_D' &= (1 - P_{\text{Sleep}}) \times \lambda_D \\
\lambda_I' &= (1 - P_{\text{Sleep}}) \times \lambda_I
\end{align*}
\]

(5.13)

where \(P_{\text{Sleep}}\) is the steady state probability that the MC is in sleep and is calculated by \(E[\text{mark}(\text{Sleep})]\) where \(E[X]\) stands for the expected value of \(X\). The last two rates are “aggregate” rates to account for the fact that an MC may be simultaneously engaged in multiple Internet or intranet sessions and they can be calculated by:

\[
\begin{align*}
\lambda_{\text{DIP}}' &= E[\text{mark}(\text{ISessions})] \times \lambda_{\text{DIP}} \\
\lambda_{\text{ILP}}' &= E[\text{mark}(\text{LSessions})] \times \lambda_{\text{ILP}}
\end{align*}
\]

(5.14)

The stochastic model underlying the SPN model shown in Fig. 5.4 is a continuous-time Markov chain. Let \(P_i\) denote the probability that the underlying Markov chain is found in a state that the current forwarding chain length is \(i\). Let \(S\) denote the set of states in the underlying Markov chain. Then \(C_{\text{location}}\) can be calculated as follows:

\[
C_{\text{location}} = \sum_S P_i C_{i, \text{location}}
\]

(5.15)

where \(C_{i, \text{location}}\) is calculated as:

\[
C_{i, \text{location}} = \begin{cases} 
2\tau & \text{if } 1 \leq i < K \\
2\alpha\tau & \text{if } i = K 
\end{cases}
\]

(5.16)

The location tracking cost \(C_{\text{tracking}}\) can be calculated as follows:

\[
C_{\text{tracking}} = \sum_S P_i C_{i, \text{tracking}}
\]

(5.17)

where \(C_{i, \text{tracking}}\) is either \(C_{i, \text{tracking}, I}\) or \(C_{i, \text{tracking}, L}\). The equations for calculating \(C_{i, \text{tracking}, I}\) and \(C_{i, \text{tracking}, L}\) are shown as follows:

\[
\begin{align*}
C_{i, \text{tracking}, I} &= (2\alpha + i) \times \tau \\
C_{i, \text{tracking}, L} &= (4\alpha + i) \times \tau
\end{align*}
\]

(5.18)

The packet delivery cost \(C_{\text{delivery}}\) is derived in a similar way as follows:

\[
C_{\text{delivery}} = \sum_S P_i C_{i, \text{delivery}}
\]

(5.19)
where $C_{i,\text{delivery}}$ is either $C_{i,\text{delivery},I}$ or $C_{i,\text{delivery},L}$. $C_{i,\text{delivery},I}$ and $C_{i,\text{delivery},L}$ can be calculated as follows:

\[
C_{i,\text{delivery},I} = (\alpha + i) \times \tau \\
C_{i,\text{delivery},L} = \beta \tau
\]  

(5.20)

Intranet sessions in WMNs, which involve two peers interacting with each other bi-directionally, usually have similar packet arrival rates in both directions. It indicates that the location information of each peer stored by the serving MR of the other peer is updated in a similar rate. Thus, data packets sent and received between the two peers usually travel the same distance $\beta$ on the average. The delivery cost of intranet data packets denoted by $C_{\text{delivery},L}$ is therefore $\beta \tau$.

As analyzed above when deriving the transition rate of transition $\text{LocUpdate}$, $C_{\text{reconnection}}$ can be derived as follows:

\[C_{\text{reconnection}} = 2\alpha \tau \]  

(5.21)

The computational procedure outlined above can be easily implemented by associating the SPN model with reward functions and calculating the steady-state rewards, using the SPNP [15] package.

### 5.3 Performance Analysis

In this section, we analyze the performance of LMMesh, in terms of the total communication cost incurred per time unit. We also carry out a comparative performance study to compare LMMesh with MIP-RR [44], a tunnel-based scheme, WMM [6], a pure routing-based scheme, and a pure pointer forwarding scheme called the dynamic anchor scheme [70]. In the following, SMR is defined as $\text{SMR} = \frac{\lambda_{\text{DIP}} + \lambda_{\text{UIP}}}{\sigma}$. Unless explicitly stated, $\lambda_{\text{DIP}}$ and $\lambda_{\text{UIP}}$ are fixed, while $\sigma$ is varied, i.e., SMR is inversely proportional to $\sigma$. The value of SMR varies from 8 to 256 in the analysis to account for the diversity of MCs in terms of service and mobility characteristics, and meanwhile ensures that the results are reasonably clear and representative. Table 5.2 lists the parameters and their default values used in the performance evaluation. The values of $\gamma$ and $\delta$ are chosen in accordance with the assumptions made in Chapter 3. The default value of $\zeta$ is chosen to be 10 because it is observed that the average ratio of the traffic load on the downlink to that on the uplink is 10 in web services [47]. The unit of time is second. The values of $P_q$ and $P_r$ are chosen according to their representative values presented in [6] and [18], respectively. All costs presented below are normalized with respect to $\tau = 1$. 
Table 5.2: Parameters and their default values.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Parameter</th>
<th>Value</th>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\gamma$</td>
<td>100</td>
<td>$\delta$</td>
<td>10</td>
<td>$\zeta$</td>
<td>10</td>
</tr>
<tr>
<td>$\lambda_{UI}$</td>
<td>600</td>
<td>$\lambda_{DI}$</td>
<td>600</td>
<td>$\mu_I$</td>
<td>300</td>
</tr>
<tr>
<td>$\tau$</td>
<td>1</td>
<td>$\alpha$</td>
<td>30</td>
<td>$\beta$</td>
<td>30</td>
</tr>
<tr>
<td>$\omega_w$</td>
<td>900</td>
<td>$\omega_s$</td>
<td>1800</td>
<td>$P_f/P_b$</td>
<td>1/4</td>
</tr>
<tr>
<td>$P_g$</td>
<td>10.0%</td>
<td>$P_q$</td>
<td>5.0%</td>
<td>$P_r$</td>
<td>50.0%</td>
</tr>
</tbody>
</table>

Figure 5.5: Cost vs. $K$.

5.3.1 Performance Evaluation of LMMesh

Fig. 5.5 shows $C_{LMMesh}$ as a function of $K$, under different SMRs. The figure demonstrates that there exists an optimal threshold $K_{\text{optimal}}$ that minimizes $C_{LMMesh}$. It can be observed in the figure that $C_{LMMesh}$ increases as SMR decreases. This is because as $\sigma$ increases (recall that SMR is inversely proportional to $\sigma$), the signaling cost incurred by location management increases, and, consequently, $C_{LMMesh}$ increases as well.

Figure 5.6: $K_{\text{optimal}}$ vs. $SMR$. 
Fig. 5.6 plots $K_{optimal}$ as a function of SMR. It can be observed in the figure that $K_{optimal}$ decreases as SMR increases. The reason is that as $\sigma$ becomes lower, allowing a shorter forwarding chain is favorable in order to reduce the packet delivery cost and thus the total communication cost.

![Figure 5.7: Cost vs. $\zeta$.](image1)

Fig. 5.7 illustrates $C_{LMMesh}$ as a function of $\zeta$, under different SMRs. $\zeta$ is a critical system parameter as it largely determines the rate of data packet arrivals at the gateway, and accordingly the rate of location update by data packets. As can be seen in the figure, $C_{LMMesh}$ increases monotonically as $\zeta$ increases. Given a fixed downlink Internet packet arrival rate $\lambda_{DIP}$, the rate of data packet arrival at the gateway and accordingly the rate of location update by data packets decrease as $\zeta$ increases. The result is that the expected steady-state forwarding chain length and consequently $C_{LMMesh}$ increase as $\zeta$ increases. This is justified by Fig. 5.8, which shows the steady-state average forwarding chain length as a function of $\zeta$.

![Figure 5.8: Average forwarding chain length vs. $\zeta$.](image2)

Fig. 5.9 investigates the effect of the active ratio on the performance of LMMesh. The active
ratio of an MC is defined as $\omega_w/\omega_s$. In the analysis, $\omega_s$ is fixed, while $\omega_w$ is varied, i.e., the active ratio is proportional to $\omega_w$. It can be observed in Fig. 5.9 that $C_{LMMesh}$ increases monotonically with increasing active ratio. This is because a larger $C_{LMMesh}$ is incurred as an MC spends more time in active mode.

![Figure 5.9: Cost vs. $\omega_w/\omega_s$.](image)

Fig. 5.10 plots $K_{optimal}$ as a function of the active ratio, under different SMRs. As illustrated in the figure, $K_{optimal}$ is a monotonic function of the active ratio. The reason is that as $\omega_w$ increases, the rate of forwarding chain reset due to location tracking increases. Thus $K_{optimal}$ increases accordingly to ensure that $C_{LMMesh}$ is minimized.

![Figure 5.10: $K_{optimal}$ vs. $\omega_w/\omega_s$.](image)

5.3.2 Performance Comparison

In this section, we compare LMMesh with MIP-RR [44], WMM [6], and the dynamic anchor scheme [70]. Here we first note that the extra cost of informing a MC’s new GFA to all Intranet and Internet CNs upon a gateway-level handoff would be the same for all protocols. Also for Intranet
traffic resulting from two MCs in two separate gateway zones, the extra cost for the source MC to route packets to the destination MC through the destination MC’s gateway would be the same for all protocols. Consequently, it suffices to compare protocol performance based on the cost incurred while an MC is within a gateway zone. MIP-RR is a micro-mobility management scheme that aims at reducing the global location handoff signaling overhead and latency by performing location registrations locally within the service region of a regional mesh router (RMR). Specific to the use of MIP-RR in WMNs, each RMR runs on an MR and handles location changes of MCs locally within its service region. Whenever an MC moves to a new MR within the fixed service region of a RMR, it informs the RMR of its location change. When the MC moves from the service region of its current RMR to that of a new RMR, it informs the gateway and all its intranet correspondence nodes of the change in RMR. Because the service region of a RMR is fixed, we can use a threshold \( D \) of the distance between the RMR and any MR within its service region to model its service region boundary. An MC is considered moving outside of the service region of its RMR when the distance between its serving MR and the RMR exceeds the threshold. The dynamic anchor scheme is a pure pointer forwarding scheme that dynamically determines the optimal threshold of the forwarding chain length. WMM is a pure routing-based mobility management scheme with opportunistic location updates through packet routing. We demonstrate that LMMesh outperforms all three schemes. These two schemes are chosen to demonstrate the benefit of integrating routing-based location update and pointer forwarding by LMMesh. We demonstrate that LMMesh outperforms all three schemes. It is worth emphasizing that because the total communication cost is on a per time unit bases, even a small performance gain of 5% to 10% will be significant over time.

The total communication cost incurred per time unit by WMM consists of the signaling cost of location handoff operations, the signaling cost of location queries upon reconnection, and the packet delivery cost. Let \( C_{wmm} \) denote the total communication cost incurred per time unit by WMM. Then \( C_{wmm} \) is calculated as:

\[
C_{wmm} = C_{\text{location}} \times \sigma' + C_{\text{delivery},I} \times \lambda'_{\text{DIP}} + C_{\text{delivery},L} \times \lambda'_{\text{ILP}} + P_q \times C_{\text{query}} \times \omega
\]  

(5.22)

Because LMMesh and WMM essentially share the same characteristics for packet routing, the equations for calculating \( C_{\text{delivery},I} \) and \( C_{\text{delivery},L} \) are the same as those presented in Section 5.2.3. In WMM, the signaling cost of the location registration procedure is calculated as: \( C_{\text{location}} = 2\tau \). When an MC’s current serving MR is unknown in WMM, a location query procedure is executed by the gateway by broadcasting a route request message to all MRs. The current serving MR of
the MC replies to the gateway a route response message. The signaling cost of the location query procedure denoted by $C_{\text{query}}$ is therefore the sum of the cost of broadcasting the route request message and the cost of transmitting the route response message, which is $\alpha \tau$.

We define the cost of broadcasting a route request message as the number of broadcasts required to deliver the message to all MRs, instead of the sum of one-hop transmission costs, because of the broadcasting nature of wireless transmission. We assume that a flooding algorithm based on self-pruning [18] is used for broadcasting in WMNs. Using such an algorithm, each node will rebroadcast a flooding packet no more than once. Thus, the number of broadcasts required to deliver the message to all MRs, can be calculated as: $P_r \times N_{MR}$. Therefore, we have $C_{\text{query}} = P_r \times N_{MR} + \alpha \tau$.

We assume the square-grid mesh network model for WMNs. In such a mesh network, the average distance between two arbitrary nodes, denoted by $\beta$, can be derived using the approach proposed in [48], given the dimension of the mesh network. It indicates that we can obtain the dimension of the mesh network by reversely applying the approach, given $\beta$. The detailed calculations are shown as follows:

$$\beta = \frac{2M}{3} \Rightarrow N_{MR} = M^2 = \left(\frac{3\beta}{2}\right)^2 \quad (5.23)$$

where $M$ denotes the dimension of the mesh network.

For the dynamic anchor scheme, the total communication cost incurred per time unit can be expressed using Equation 5.12, with the additional cost for an MC to inform its CNs when its location information stored in the location database is updated and its forwarding chain is reset. Accordingly, some equations presented in Section 5.2.3 need to be revised as follows:

$$C_{i,\text{location}} = \begin{cases} 2\tau & \text{if } 1 \leq i < K \\ (2\alpha + 2N_L\beta) \times \tau' & \text{if } i = K \end{cases} \quad (5.24)$$

$$C_{i,\text{tracking},I} = (2\alpha + 2N_L\beta + i) \times \tau$$

$$C_{i,\text{tracking},L} = (4\alpha + 2N_L\beta + i) \times \tau$$

$$C_{i,\text{delivery},L} = (\beta + i) \times \tau$$

where $N_L$ denotes the number of active intranet correspondence nodes.

MIP-RR is essentially equivalent to the dynamic anchor scheme with a fixed threshold $D$ and without the location tracking mechanism. Additionally, because the location handoff/update operations in MIP-RR are different from those in the dynamic anchor scheme, which is a pointer forwarding scheme, the equation shown below is different from the one above.

$$C_{i,\text{location}} = \begin{cases} 2i\tau' & \text{if } 1 \leq i < D \\ (2\alpha + 2N_L\beta) \times \tau' & \text{if } i = D \end{cases} \quad (5.26)$$
where $i$ denotes the distance between the MC’s current serving MR and its GFA in this case, and $\tau'$ is calculated as $\tau' = (1 + \epsilon) \times \tau$, where $\epsilon$ denotes the percentage of increase in $\tau$ due to the additional IP encapsulation/decapsulation overhead in tunnel-based schemes. Below we let $D = 4$ to evaluate the performance of MIP-RR.

![Figure 5.11: Performance comparison: cost vs. SMR.](image1)

![Figure 5.12: Performance comparison: average forwarding chain length vs. SMR.](image2)

Fig. 5.11 compares the total communication cost incurred per time unit by the four schemes as a function of SMR. As can be seen in the figure, LMMesh significantly outperforms the other three schemes, namely, WMM (labeled as “routing-based”), dynamic anchor (labeled as “pointer forwarding”), and MIP-RR (labeled as “tunnel-based”), especially when SMR is small. The advantage of LMMesh is due to the combination of routing-based location information update and pointer forwarding which eliminates the problem of opportunistic location updates. As expected, tunnel-based MIP-RR shows the worst performance because of its use of a rigid GFA service region size for all MCs that leads to suboptimal overall performance and the extra overhead introduced by packet encapsulation/decapsulation. Fig. 5.12 compares the average forwarding chain length as
a function of SMR among the four schemes. As can be seen in Fig. 5.12, WMM due to its opportunistic nature has a much larger forwarding chain length than the dynamic anchor scheme and LMMesh over a wide range of SMR. Because a larger forwarding chain length means a higher per time unit packet delivery cost, this figure explains why WMM performs worse than the dynamic anchor scheme and LMMesh.

![Figure 5.13: Performance comparison: cost vs. $\zeta$.](image)

![Figure 5.14: Performance comparison: average forwarding chain length vs. $\zeta$.](image)

Fig. 5.13 compares the total communication cost incurred per time unit by two routing-based schemes, namely, LMMesh and WMM, as a function of $\zeta$. As expected, the total communication cost increases monotonically with increasing $\zeta$ in both schemes. However, $C_{wmm}$ increases much faster than $C_{LMMesh}$. This indicates that the impact of the rate of data packet arrivals at the gateway on the total communication cost is much more significant in WMM than in LMMesh because location information is updated primarily by uplink Internet data packets in WMM. This observation is well supported by Fig. 5.14, which compares the average forwarding chain length as a function of $\zeta$ between LMMesh and WMM. As can be seen in the figure, the average forwarding
chain length of WMM increases much faster than that of LMMesh with increasing $\zeta$.

![Figure 5.15: Performance comparison: cost vs. $\omega_w/\omega_s$.](image)

Fig. 5.15 compares the total communication cost incurred per time unit as a function of the active ratio among the four schemes. As expected, the total communication cost incurred by each of the four schemes increases monotonically as the active ratio increases. LMMesh again outperforms the other three schemes.

![Figure 5.16: Performance comparison: cost vs. $\lambda_I$.](image)

Fig. 5.16 and Fig. 5.17 compare the total communication cost incurred per time unit by the four schemes, as a function of $1/\lambda_I$ and $1/\mu_I$, respectively. It can be seen in the figures that the total communication cost incurred per time unit by each of the four schemes decreases monotonically with decreasing $\lambda_I$, whereas it increases monotonically with decreasing $\mu_I$. This is because when $\mu_I$ is fixed and $\lambda_I$ decreases, the average number of on-going sessions of each MC decreases accordingly. Conversely, when $\lambda_I$ is fixed and $\mu_I$ decreases, the average number of on-going sessions of each MC increases. As shown in both figures, LMMesh outperforms the other three schemes.
5.3.3 Sensitivity Analysis

In this section, we investigate the sensitivity of the above analytical results with respect to the network coverage model assumed. Specifically, in the following analysis, a hexagonal network coverage model is used instead of the square-grid mesh model. In the hexagonal network coverage model, the coverage area of each MR is called a cell, and each MR has six direct neighbors. An MC can move randomly from an MR to one of its direct neighbors with the same probability. Thus, \( P_f = \frac{5}{6} \) and \( P_b = \frac{1}{6} \), under the hexagonal network coverage model.

Fig. 5.18 illustrates \( C_{LMMesh} \) as a function of \( K \), under the hexagonal network coverage model. It can be observed that cost curves shown in this figure and Fig. 5.5 exhibit high similarity in shape. The same conclusion can be drawn by comparing Fig. 5.19 with Fig. 5.11. Based on these observations, we can draw the conclusion that analytical results obtained are valid and are not sensitive to the network coverage model.
5.4 Summary

In this chapter, we proposed and analyzed a routing-based location management scheme with pointer forwarding, namely LMMesh, for wireless mesh networks. LMMesh integrates routing-based location update and pointer forwarding into a single scheme that exploits the advantages of both methods, while avoiding their drawbacks. LMMesh integrates these two methods to offer a complete solution to location management in wireless mesh networks, and considers the effect of the integration on the overall network cost incurred by location management and packet delivery. The tradeoff between the signaling cost for location management and the service cost for packet delivery is explored by LMMesh by dynamically determining the optimal threshold for the forwarding chain length that minimizes the overall network cost. LMMesh is optimal on a per-user basis and is adaptive to the changing mobility and service behaviors of an MC, as the optimal forwarding chain length is determined specifically for the MC, dynamically based on the specific mobility and service characteristics of the MC. LMMesh can be used in either single-gateway or multi-gateway WMNs, and it is scalable as the mobility management role is dynamically shared among the gateways and among the MR such that no single gateway or MR would become a bottleneck.

We developed an analytical model based on stochastic Petri net techniques to analyze the performance of LMMesh and performed a comparative study to compare LMMesh against a tunnel-based location management scheme, a pure routing-based scheme, and a pure pointer forwarding scheme. Our results demonstrated that LMMesh is consistently superior to these existing schemes for location management in WMNs. We attribute the superiority of LMMesh to the integration of routing-based location update and pointer forwarding.
Chapter 6

Integrated Mobility and Cache Consistency Management for Mobile Data Access

In this chapter\(^1\), we propose and analyze an integrated mobility and cache consistency management scheme, named Adaptive Per-user Per-object Cache Consistency Management (APPCCM), for efficiently supporting mobile data access in WMNs. APPCCM provides two data access and caching modes: a data object can be cached either directly at the MC, or at a *data proxy* running on an MR dynamically selected by APPCCM. We use the terms Client-Cache Mode (CCM) and Data-Proxy Mode (DPM) to refer to these two modes. CCM is based on existing asynchronous stateful-based cache invalidation schemes [58,68] augmented with the capability of integrated cache consistency and mobility management such that MC can perform data access, data caching and cache consistency management while roaming in a WMN. DPM is distinct from traditional approaches by exploiting the capability of increasingly powerful MRs to perform data caching in addition to routing.

APPCCM is based on a stateful approach by which when a data object is updated, an IR is asynchronously sent from the server to the MC and data proxies that keep a cached copy of the data object. Recall that the server is stateful as it keeps state information about which clients cache which data objects. Specific to this work, the server sends an IR to the gateway whenever it updates a data object, and the gateway forwards the IR to those MC and data proxies that keep a cache copy of the data object. Therefore, the gateway rather than the server is stateful and keeps information about where data objects are cached. We assume that the gateway always keeps a copy of every data object accessed by MC within the WMN. Therefore, the gateway is like a replica of

\(^1\)Part of this chapter is published as a journal paper in Journal of Parallel and Distributed Computing [57].
the server.

We develop an analytical model based on SPN for evaluating the performance of APPCCM, given parameters characterizing the MC’s mobility and data query/update characteristics, and the WMN conditions. We demonstrate via both model-based analysis and simulation validation that APPCCM significantly outperforms non-adaptive cache consistency management schemes that always cache data objects at the mesh client, or at the mesh client’s current serving mesh router for mobile data access in wireless mesh networks.

The rest of the chapter is organized as follows. The proposed APPCCM scheme is presented in Section 6.1. In Section 6.2, we develop mathematical models based on stochastic Petri net for evaluating APPCCM. Performance analysis and comparison are carried out in Section 6.3. The chapter is summarized in Section 6.4.

### 6.1 Adaptive Per-User Per-Object Cache Consistency Management

#### 6.1.1 DPM versus CCM

There are two caching modes in APPCCM, namely CCM and DPM. In CCM, a data object accessed by an MC is cached directly by the MC, whereas in DPM, the data object is cached by a data proxy running on an MR. A data proxy is essentially a data cache maintained by an MR. Modern MRs have sufficient computing power and storage capacity to perform both routing and data caching [54]. The rationale of using data proxies to cache data objects is that it incurs less network cost than always caching data objects directly at the MC, under certain circumstances. More specifically, when a data object is updated more frequently than being accessed by an MC such that the invalidation cost is dominating, it may be beneficial to cache the data object at a data proxy rather than locally at the MC to reduce the invalidation cost and hence the total communication cost. On the other hand, if a data object is accessed more frequently by the MC than being updated such that the access cost is dominating, it may be beneficial to let the MC cache the data object directly to avoid the additional cost of accessing a data proxy. Therefore, there exists a tradeoff between the access cost and invalidating cost. APPCCM exploits this tradeoff and adaptively decides on a per-user per-object basis where to cache a data object based on the data object’s QUR and the MC’s mobility characteristic. The decision is made independently for each data object accessed by each individual MC. Therefore APPCCM is an adaptive per-user per-object cache consistency management scheme.
Table 6.1: Fields of a caching status table entry.

<table>
<thead>
<tr>
<th>Field</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object ID</td>
<td>The data object identifier; each data object has a unique identifier</td>
</tr>
<tr>
<td>Caching location</td>
<td>The location where the data object is cached</td>
</tr>
<tr>
<td>Address of data proxy</td>
<td>The address of the data proxy where the data object is cached if applicable</td>
</tr>
<tr>
<td>Time stamp</td>
<td>The time stamp when the cached copy of the data object is most recently updated due to query</td>
</tr>
<tr>
<td>Validity</td>
<td>A flag indicating whether the cached copy is valid or not</td>
</tr>
</tbody>
</table>

6.1.2 Data Access and Caching

In APPCCM, each MC maintains a caching status table that keeps the caching status of each data object it has accessed. A caching status table entry has five fields as shown in Table 6.1. When an MC receives a new data query from an application, it first checks its caching status table to see whether an entry for the queried data object exists or not in the table. If an entry is found, it determines where the data object is currently cached and if the cached copy is still valid. An MC can determine whether the cached copy of a data object is valid by checking the validity flag in the caching status table. The validity flag of the cached copy of a data object is set to false when it is invalidated by an IR, and is set to true when the data object is queried and the cached copy is updated. Depending on the result of this table lookup, the query is answered accordingly in different ways. The pseudo code presented below describes the query processing algorithm.

We consider that in addition to the server, an MC can also update any data object for which it keeps a cached copy. Therefore, MC has both read and write permissions. Whenever an MC updates a data object, it sends the updated object and an IR to the gateway, which forwards the updated data object to the server and, upon receiving an acknowledgment from the server that the update is accepted, forwards the IR to those MC and data proxies that keep a cached copy of the data object to be invalidated.

Fig. 6.1 illustrates examples of query and update processing in APPCCM. As the figure shows, MC1 that is connected to MR1 employs MR2 as a data proxy to cache some data objects it has accessed. Upon receiving a query for a data object cached in the data proxy running on MR2, MC1 sends the query to MR2, which sends the queried data object back to MC1. In another example, MC2 updates a data object cached in the data proxy running on MR3. After the update is completed, MR3 sends the updated object and an IR to the gateway, which then forwards the
Algorithm 1: The query processing algorithm.

```
if an entry is not found then
    the MC sends the query to the server to retrieve a fresh copy of the data object;
    if CCM is to be used to cache the object then
        the MC puts the received data object into its local cache upon receiving it;
    else
        upon receiving the data object, the MC’s current serving MR puts it into the data proxy
        before forwarding it to the MC;
    end if
    the MC updates its caching status table;
else
    if the data object is found cached by the MC then
        if the cached copy is still valid then
            the query is answered immediately locally;
        else
            the MC sends the query to the server, and upon receiving the data object, the MC
            updates the cached copy and the caching status table;
        end if
    else
        the MC sends the query to the data proxy specified in the caching status table;
        if the cached copy is still valid then
            the data proxy sends the data object to the MC;
        else
            the data proxy forwards the query to the server, and upon receiving the data object,
            updates the cached copy and forwards the data object to the MC;
        end if
    end if
    upon receiving the data object, the MC updates the caching status table;
end if
```
updated data object to the server. Upon receiving an acknowledgment from the server that the update is accepted, the gateway forwards the IR to those MC and data proxies that keep a cached copy of the data object being invalidated, namely, MC3 and MR4 in this example.

6.1.3 MC Mobility and Data Migration

We explicitly consider the effect of mobility of MC on data caching and cache consistency management in APPCCM. MC mobility affects the two caching modes of APPCCM differently. Specifically, in DPM, a threshold denoted by $K$ is specified for each data object accessed by an MC, such that when the distance between the MC’s current serving MR and the data proxy where the data object is cached reaches $K$ due to MC mobility, the data object is migrated to the data proxy on the MC’s current serving MR. The optimal threshold $K_{optimal}$ that results in minimized total communication cost is determined dynamically on a per-user per-object basis.

DPM can be easily implemented, as illustrated by Fig. 6.2. Specifically, each MC keeps a counter for each data object that records the number of location changes of the MC since the data object’s most recent migration. Each time when the MC moves and changes its serving MR, the counter is incremented by 1. When the counter reaches the threshold $K$ after a movement, the data object will be migrated from its current data proxy to the one running on the new serving
MR of the MC. After the data migration, the counter is reset to zero.

In CCM, a location management scheme is employed to track the locations of MC in order for the gateway to deliver IRs to the destination MC. We develop a per-user location management scheme for CCM based on pointer forwarding [35,70]. In this scheme, the gateway maintains a location database where the address of the forwarding chain head of each MC is kept. A threshold denoted by \( L \) is specified for each MC to regulate the allowable forwarding chain length. The optimal threshold \( L_{optimal} \) under which the overall communication cost for location management and service delivery is minimized is dynamically determined for each MC based on the MC’s mobility and service characteristics. When an MC moves and changes its serving MR, a forwarding pointer is setup between the two involved MRs, and the forwarding chain length is increased by 1. When the forwarding chain length of the MC reaches the threshold \( L \), its current forwarding chain is reset and the new serving MR becomes its new forwarding chain head. A location binding update message is sent to the gateway in this case to update the location information of the MC, i.e., the address of its forwarding chain head.

Fig. 6.3 illustrates how the location management scheme in CCM is implemented. Like in DPM, each MC maintains a counter that records the number of movements since the most recent location update. Each time the MC moves to a new MR, the counter is incremented by 1, indicating that the length of the forwarding chain increases by 1. When the counter reaches the threshold \( L \) after
a movement, a location update is performed and the MC’s current forwarding chain is reset. After the location update, the new serving MR becomes its new forwarding chain head, and the counter is reset to zero.

6.2 Analytical Modeling

Table 6.2: Parameters used in performance modeling and analysis.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Physical meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>σ</td>
<td>Input</td>
<td>Mobility rate of an MC</td>
</tr>
<tr>
<td>λ</td>
<td>Input</td>
<td>Rate of queries from an MC for a data object</td>
</tr>
<tr>
<td>μ</td>
<td>Input</td>
<td>Rate of updates of the server to a data object</td>
</tr>
<tr>
<td>δ</td>
<td>Input</td>
<td>Aggregate rate of updates of all MC but the one under consideration to a data object</td>
</tr>
<tr>
<td>η</td>
<td>Input</td>
<td>Rate of updates of the MC under consideration to a data object</td>
</tr>
<tr>
<td>ω</td>
<td>Derived</td>
<td>Rate of reconnection when an MC switches from idle mode back to active mode</td>
</tr>
<tr>
<td>ω_w</td>
<td>Input</td>
<td>( \frac{1}{\omega_w} ) indicates the average duration of disconnection of the MC before a transition from idle mode to active mode</td>
</tr>
<tr>
<td>ω_s</td>
<td>Input</td>
<td>( \frac{1}{\omega_s} ) denotes the average duration within which the MC keeps connected before a transition from active mode to idle mode</td>
</tr>
<tr>
<td>QUR</td>
<td>Derived</td>
<td>Query to update ratio, defined as ( \frac{\lambda}{\mu+\delta+\eta} )</td>
</tr>
<tr>
<td>QMR</td>
<td>Derived</td>
<td>Query to mobility ratio, defined as ( \frac{\lambda}{\sigma} )</td>
</tr>
<tr>
<td>P active</td>
<td>Derived</td>
<td>Probability of an MC in active mode, defined as ( \frac{\omega_w}{\omega_w+\omega_s} )</td>
</tr>
<tr>
<td>α</td>
<td>Input</td>
<td>Average distance (number of hops) between the gateway and an arbitrary MR</td>
</tr>
<tr>
<td>β</td>
<td>Input</td>
<td>Average distance (number of hops) between the current serving MR of an MC after it reconnects and its serving MR before disconnection</td>
</tr>
<tr>
<td>τ</td>
<td>Input</td>
<td>One-hop communication cost between two MRs</td>
</tr>
<tr>
<td>P hit</td>
<td>Derived</td>
<td>Cache hit ratio</td>
</tr>
<tr>
<td>P miss</td>
<td>Derived</td>
<td>Cache miss ratio</td>
</tr>
</tbody>
</table>

In this section, we develop analytical models based on stochastic Petri net (SPN) for analyzing the performance of APPCCM. Table 6.2 lists the parameters used in performance modeling and analysis.

6.2.1 SPN Models for APPCCM

Fig. 6.4 presents the SPN model for DPM. The meanings of places and transitions in the SPN model are defined in Table 6.3. \( \text{Mark}(P) \) returns the number of tokens in place \( P \). In the SPN model, \#(P) associated with an arc means that the multiplicity of the arc is equal to the number
Figure 6.4: The SPN model for DPM.

Table 6.3: The meanings of places and transitions defined in the SPN model for DPM.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Move</td>
<td>A timed transition modeling MC movement</td>
</tr>
<tr>
<td>Moves</td>
<td>$\text{Mark}(\text{Moves})$ represents the number of movements</td>
</tr>
<tr>
<td>DataMigration</td>
<td>A timed transition modeling the migration of the data object between two data proxies</td>
</tr>
<tr>
<td>Active2Idle</td>
<td>A timed transition modeling the state transition of the MC from active mode to idle mode</td>
</tr>
<tr>
<td>Idle</td>
<td>$\text{Mark}(\text{Idle})=1$ means the MC is in idle mode</td>
</tr>
<tr>
<td>Idle2Active</td>
<td>A timed transition modeling the state transition of the MC from idle mode to active mode</td>
</tr>
<tr>
<td>Active</td>
<td>$\text{Mark}(\text{Active})=1$ means the MC is in active mode</td>
</tr>
<tr>
<td>StatusChecking</td>
<td>A timed transition modeling the event of checking the caching status after the MC reconnects</td>
</tr>
</tbody>
</table>
of tokens in place $P$. In Fig. 6.4 we put in numbers in parenthesis to label the SPN model sequence below. The SPN model for DPM is constructed as follows:

- The event of MC movement is modeled by transition $Move$, the transition rate of which is $\sigma$. When an MC moves to and is associated with a new MR, a token is put into place $Moves$, which represents the number of times the MC has moved since the most recent migration of the data object under consideration.

- When the number of movements since the last data object migration reaches $K$, i.e., the number of tokens in place $Moves$ is accumulated to $K$, transition $DataMigration$ is enabled and fired, representing the event of migrating the data object from the data proxy where it is currently cached to the one running on the new serving MR.

- An MC typically switches alternatively between active mode and idle mode during its stay in a WMN. Initially the MC is in active mode, and can send and receive packets. After staying in active mode for a period of time, the MC is switched to idle mode to save battery life. This is modeled by transition $Active2Idle$, the transition rate of which is $\omega_s$. After transition $Active2Idle$ is fired, a token is put into place $Idle$, representing that the MC is switched to idle mode. The MC reconnects to the WMN after being in idle mode for some time. This is modeled by transition $Idle2Active$, the transition rate of which is $\omega_w$.

- When the MC reconnects, it sends a query message to the data proxy where the data object under consideration is cached to check its caching status. This event is modeled by transition $StatusChecking$. If the cached copy is still valid, it is migrated to the new data proxy running on the MC’s current serving MR; otherwise, the data proxy simply responses with a message telling that the cached copy is obsolete. After the status checking, the number of movements since the last data migration is reset to zero, i.e., all tokens in place $Moves$ are consumed by transition $StatusChecking$.

Fig. 6.5 depicts the SPN model for CCM. Table 6.4 defines the meanings of places and transitions in the model. In Fig. 6.5 we put in numbers in parenthesis to label the SPN model sequence below. The SPN model for CCM is constructed as follows:

- The event of MC movement is modeled by transition $Move$, the transition rate of which is $\sigma$. When an MC moves to and is associated with a new MR, a token is put into place $tmp$,
Figure 6.5: The SPN model for CCM.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Move</td>
<td>A timed transition modeling MC movement</td>
</tr>
<tr>
<td>tmp</td>
<td>Mark(tmp) = 1 means that the MC just moves to a new MR</td>
</tr>
<tr>
<td>AddPointer</td>
<td>A timed transition modeling setting up a forwarding pointer between two neighboring MRs</td>
</tr>
<tr>
<td>FL</td>
<td>Mark(FL) returns the MC’s current forwarding chain length</td>
</tr>
<tr>
<td>Reset</td>
<td>A timed transition modeling a location update event that resets the forwarding chain</td>
</tr>
<tr>
<td>Active2Idle</td>
<td>A timed transition modeling the state transition of the MC from active mode to idle mode</td>
</tr>
<tr>
<td>Idle</td>
<td>Mark(Idle) = 1 means the MC is in idle mode</td>
</tr>
<tr>
<td>Idle2Active</td>
<td>A timed transition modeling the state transition of the MC from idle mode to active mode</td>
</tr>
<tr>
<td>Active</td>
<td>Mark(Active) = 1 means the MC is in active mode</td>
</tr>
<tr>
<td>AfterReconnection</td>
<td>A timed transition modeling the event of retrieving IRs from the forwarding chain head received during the MC’s disconnection and sending the gateway a location binding update message after the MC reconnects</td>
</tr>
</tbody>
</table>
which subsequently enables and fires transition \textit{AddPointer}, meaning that a new forwarding pointer is setup between the old and new serving MR. After transition \textit{AddPointer} is fired, a token is put into place \textit{FL} that represents the length of the forwarding chain, indicating that the forwarding chain length is increased by 1.

- When the length of the forwarding chain reaches \( L \), i.e., the number of tokens in place \textit{FL} is accumulated to \( L \), transition \textit{Reset} is enabled and fired, representing that the current forwarding chain is reset and the new current serving MR becomes the MC’s new forwarding chain head.

- Transitions \textit{Active2Idle} and \textit{Idle2Active} represent the same physical meanings as in the SPN model for DPM.

- When the MC reconnects, it sends a query message to its current forwarding chain head to retrieve any IRs received by the forwarding chain head during its disconnection. The MC also sends a location binding update message to the gateway to update its location information, i.e., the address of the forwarding chain head. After the location update, the current serving MR becomes the MC’s new forwarding chain head. Transition \textit{AfterReconnection} models the above events performed by the MC after it reconnects.

### 6.2.2 Parameterization

Transition \textit{DataMigration} in the SPN model for DPM represents the event of migrating a data object between two data proxies when the threshold \( K \) with respect to the data object is reached. In this event, the MC that initiates the migration sends a data migration request to the data proxy where the data object is cached, asking it to migrate the data object to the data proxy on the MC’s new serving MR. After the data migration is completed, the new serving MR sends a data migration acknowledgment to the MC. The signaling cost incurred by the event is \( 2K\tau + 2\tau \) because the distance between the two data proxies is \( K \) hops. Therefore, the transition rate of \textit{DataMigration} denoted by \( \mu_{DataMigration} \) is calculated as:

\[
\mu_{DataMigration} = \frac{1}{2K\tau + 2\tau} \quad (6.1)
\]

Transition \textit{StatusChecking} in the SPN model for DPM represents the event of checking the caching status of a data object in a data proxy after the MC that initiates the status checking reconnects, and if the data object is still valid, migrating it to the data proxy on the MC’s new
serving MR. Therefore, the signaling cost for status checking upon reconnection is the sum of the

cost for sending the status checking request and the cost for migrating the data object if it is

still valid or for transmitting the IR if the data object has already been invalidated. The cost is

\(2\beta\tau + 2\tau\) because the distance between the current serving MR of the MC after it reconnects and

its serving MR before disconnection is \(\beta\) hops. The transition rate of StatusChecking denoted by

\(\mu_{\text{StatusChecking}}\) is therefore calculated as:

\[
\mu_{\text{StatusChecking}} = \frac{1}{2\beta\tau + 2\tau} \quad (6.2)
\]

Transition Reset in the SPN model for CCM represents the event of resetting the current

forwarding chain of an MC. In this event, the MC sends a location binding update message to the
gateway to update its location information, i.e., the address of the new forwarding chain head. The
gateway responds with a location binding update confirmation. The signaling cost incurred in this

event is \(2\alpha\tau + 2\tau\). Therefore, the transition rate denoted by \(\mu_{\text{Reset}}\) can be derived as follows:

\[
\mu_{\text{Reset}} = \frac{1}{2\alpha\tau + 2\tau} \quad (6.3)
\]

Transition AfterReconnection in the SPN model for CCM represents two events after an MC

reconnects. Specifically, the MC sends a query message to its forwarding chain head before dis-

cconnection to retrieve any IRs received by the head during its reconnection. The MC also sends a

location binding update message to the gateway to update its location information. The signaling

cost incurred is \(2(\alpha + \beta)\tau + 4\tau\). Therefore, the transition rate denoted by \(\mu_{\text{AfterReconnection}}\) can be

derived as follows:

\[
\mu_{\text{AfterReconnection}} = \frac{1}{2(\alpha + \beta)\tau + 4\tau} \quad (6.4)
\]

6.2.3 Performance Metrics

We use the total communication cost incurred per time unit as the metric for performance evalua-
tion. For DPM, the total communication cost incurred per time unit \(C_{DPM}\) consists of the query

cost \(C_{\text{query}}\), the invalidation cost \(C_{\text{invalidation}}\), the signaling cost for data migration \(C_{\text{migration}}\),

and the signaling cost for status checking upon reconnection \(C_{\text{reconnection}}\). Using these cost terms,

\(C_{DPM}\) is calculated as follows:
\[ C_{DPM} = \lambda' \cdot C_{\text{query}} + (\mu + \delta + \eta') \cdot C_{\text{invalidation}} + \sigma' \cdot C_{\text{migration}} + \omega \cdot C_{\text{reconnection}} \] (6.5)

In the above equation, \( \lambda' \) and \( \eta' \) denote the effective data query rate and the effective data update rate of the MC under consideration, respectively. \( \sigma' \) denotes the effective mobility rate of the MC. These are effective rates because the MC cannot access or update any data object during its disconnection. These effective rates are calculated by:

\[
\begin{align*}
\sigma' &= P_{\text{active}} \sigma \\
\lambda' &= P_{\text{active}} \lambda \\
\eta' &= P_{\text{active}} \eta
\end{align*}
\] (6.6)

Here \( P_{\text{active}} \) denotes the probability that the MC is in active mode, calculated by the ratio of the average active duration over the sum of the average active duration and the average idle duration, as follows:

\[
P_{\text{active}} = \frac{1}{\omega_s} = \frac{\omega_w}{\omega_w + \omega_s}
\] (6.7)

Below we parameterize various cost terms in Equation 6.5. We exclude the communication cost for data object or IR transmission between the server and the gateway because it is not part of the overall cost incurred to the WMN. The query cost incurred by DPM in the case of a cache hit consists of the cost for sending the query to the data proxy, and the cost for delivering the queried data object to the MC. In the cast of a cache miss, the query cost consists of the cost for sending the query to the data proxy, the cost for forwarding the query to the gateway, the cost for transmitting the data object from the gateway to the proxy, and finally the cost for delivering the queried data object to the MC. Therefore, \( C_{\text{query}} \) is calculated as (with \( n_m \) denoting the number of tokens in place \( \text{Moves} \)):

\[
C_{\text{query}} = \begin{dcases} 
2n_m \tau + 2\tau & \text{if cache hit} \\
2(\alpha + n_m) \tau + 2\tau & \text{if cache miss}
\end{dcases}
\] (6.8)

The invalidation cost incurred by DPM in the case that the MC under consideration updates a cached data object consists of the cost for sending the updated data object and IR to the data proxy, and subsequently to the gateway, and the cost for the delivery of the invalidation acknowledgment. If the invalidation is due to updates from other MC, the invalidation cost consists of the cost for pushing the IR from the gateway to the MC under consideration and the cost for transmitting the invalidation acknowledgment. Therefore, \( C_{\text{invalidation}} \) is given by (with \( \text{MC}_0 \) representing the MC under consideration):

\[
C_{\text{invalidation}} = \begin{dcases} 
2\alpha \tau & \text{if update by other MC} \\
2(n_m + \alpha) \tau + 2\tau & \text{if update by \( \text{MC}_0 \) itself}
\end{dcases}
\] (6.9)
The signaling cost for data migration in DPM is for transmitting the data migration request and acknowledgment, and for transmitting the data object between two data proxies that are \( K \) hops away from each other. \( C_{\text{migration}} \) is therefore calculated as:

\[
C_{\text{migration}} = 2K\tau + 2\tau \quad (6.10)
\]

The signaling cost for status checking upon reconnection in DPM is the sum of the cost for sending the status checking request and the cost for migrating the data object if it is still valid or for transmitting the IR if the data object has already been invalidated. Because the distance between the current serving MR of the MC after it reconnects and its serving MR before disconnection is \( \beta \) hops, \( C_{\text{reconnection}} \) is calculated as:

\[
C_{\text{reconnection}} = 2\beta\tau + 2\tau \quad (6.11)
\]

For CCM, the total communication cost incurred per time unit (\( C_{\text{CCM}} \)) consists of the query cost (\( C_{\text{query}} \)), the invalidation cost (\( C_{\text{invalidation}} \)), the signaling cost for location management (\( C_{\text{location}} \)), and the signaling cost for cache status checking upon reconnection (\( C_{\text{reconnection}} \)). Using these cost terms, \( C_{\text{CCM}} \) is calculated as:

\[
C_{\text{CCM}} = \lambda' \cdot C_{\text{query}} + (\mu + \delta + \eta') \cdot C_{\text{invalidation}} + \sigma' \cdot C_{\text{location}} + \omega \cdot C_{\text{reconnection}} \quad (6.12)
\]

The query cost incurred by CCM in the case of a cache hit is zero because the data object is retrieved locally from the cache. In the case of a cache miss, the query cost consists of the cost for sending the query to the gateway and the cost for transmitting the data object from the gateway to the MC following the forwarding chain. Therefore, \( C_{\text{query}} \) is calculated as (with \( l_f \) denoting the current forwarding chain length):

\[
C_{\text{query}} = \begin{cases} 
0 & \text{if cache hit} \\
(\alpha + l_f)\tau + \alpha\tau + 2\tau & \text{if cache miss}
\end{cases} \quad (6.13)
\]

The calculation of the invalidation cost in CCM depends on where the update to the invalidated data object originates. In the case that the update is from the server or other MC, a) if the MC under consideration is in idle mode, the cost incurred is for transmitting the IR and invalidation acknowledgment between the gateway and the MC’s forwarding chain head, and b) if the MC under consideration is in active mode, the cost incurred is for transmitting the IR and invalidation acknowledgment between the gateway and the MC. If the update is from the MC under consideration (\( MC_0 \)), the invalidation cost consists of the cost for sending the IR and the updated data
object to the gateway, and the cost for transmitting the invalidation acknowledgment. Therefore, $C_{\text{invalidation}}$ is calculated as:

$$C_{\text{invalidation}} = \begin{cases} 
2\alpha \tau & \text{if update by other MC or the server and MC}_0 \text{ is active} \\
(2\alpha + l_f)\tau + 2\tau & \text{if update by other MC or the server and MC}_0 \text{ is idle} \\
(2\alpha + l_f)\tau + 2\tau & \text{if update by MC}_0 \text{ itself}
\end{cases}$$

(6.14)

The signaling cost for location management in CCM is the cost for setting up a forwarding pointer between two neighboring MRs, if the forwarding chain length after the movement is less than the threshold $L$, or the cost for location update if the forwarding chain length after the movement reaches the threshold $L$. In the latter case, a location binding update message is sent to the gateway in this case to update the location information of the MC, i.e., the address of its forwarding chain head. Therefore, $C_{\text{location}}$ is calculated as:

$$C_{\text{location}} = \begin{cases} 
4\tau & \text{if } f_l < L \\
2\alpha \tau + 2\tau & \text{if } f_l = L
\end{cases}$$

(6.15)

The signaling cost for status checking upon reconnection in CCM consists of the cost for transmitting the status checking request and the IR if there are cached data object that have been invalidated during the disconnection, and the cost for location update. Therefore, $C_{\text{reconnection}}$ is calculated as:

$$C_{\text{reconnection}} = 2(\alpha + \beta)\tau + 4\tau$$

(6.16)

A query for a data object results in a cache hit if there is no update to the object between this query and the most recent query in the past. There will be a cache miss if there is at least one update to the object in between two consecutive queries. Therefore, the cache hit ratio of a data object denoted by $P_{\text{hit}}$ can be calculated by the average number of successive accesses that can be done during the interval between two consecutive updates, as follows:

$$P_{\text{hit}} = \frac{\lambda'}{\lambda' + \mu + \delta + \eta'}$$

(6.17)

An MC may switch between active mode and idle mode during its stay in a WMN. The average interval between two consecutive reconnections of the MC is the sum of the average active duration and the average idle duration, i.e., $\frac{1}{\omega_\text{w}} + \frac{1}{\omega_\text{s}}$. The rate of reconnection denoted by $\omega$ is therefore calculated as follows:

$$\omega = \frac{1}{\omega_\text{w} + \omega_\text{s}} = \frac{\omega_\text{w} \times \omega_\text{s}}{\omega_\text{w} + \omega_\text{s}}$$

(6.18)
Table 6.5: Parameters and their values used in performance analysis.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>QUR</td>
<td>{0.125, 0.25, 0.5, 1.0, 2.0, 4.0, 8.0}</td>
</tr>
<tr>
<td>QMR</td>
<td>{1, 2, 4, 8, 16, 32, 64, 128, 256}</td>
</tr>
<tr>
<td>(P_{active})</td>
<td>{(\frac{1}{3}, \frac{1}{5}, \frac{1}{7}, \frac{1}{9})}</td>
</tr>
<tr>
<td>(\omega_w)</td>
<td>(\frac{1}{600})</td>
</tr>
<tr>
<td>(\omega_s)</td>
<td>(\frac{1}{1200})</td>
</tr>
<tr>
<td>(\alpha)</td>
<td>{10, 20, 30}</td>
</tr>
<tr>
<td>(\beta)</td>
<td>{5, 10, 20}</td>
</tr>
<tr>
<td>(\tau)</td>
<td>1</td>
</tr>
</tbody>
</table>

6.3 Performance Analysis and Numerical Results

In this section, we analyze the performance of APPCCM, in terms of the total communication cost incurred per time unit. We also carry out a comparative performance study to compare APPCCM with non-adaptive cache consistency management schemes that always cache a data object at the MC, or at the MC’s current serving MR for supporting mobile data access in WMNs. The numerical results are obtained by first defining a SPN model as shown in Fig. 6.4 or Fig. 6.5 using SPNP [15], and then evaluating the SPN model by assigning rewards to states of the system based on the equations presented above. It is worth noting that the total communication cost is on a per time unit, per MC basis, so even a performance difference of 5-10% will be significant over time. We define a parameter called query to mobility ratio (QMR) indicating how often an MC performs data access relative to its mobility speed. The QMR of an MC with respect to a data object is defined as: QMR = \(\frac{\lambda}{\sigma}\).

Table 6.5 lists the parameters and their default values used in the performance evaluation. The time unit used is second. We select a wide range of values for QMR, QUR, and \(P_{active}\) to cover diverse data query/update, mobility and service patterns and to test their effects on performance characteristics of APPCCM. The physical meaning of \(\omega_w\) is the reciprocal of the average duration of disconnection of an MC before reconnection with the WMN. Therefore, \(\omega_w = \frac{1}{600}\) means that the MC stays disconnected for a duration of 10 minutes on average before switching to active mode and reconnecting with the WMN. Similarly, \(\omega_s = \frac{1}{1200}\) means that the MC stays in active mode for a duration of 20 minutes on average before disconnecting from the WMN and changing to idle mode. The values of \(\alpha\) and \(\beta\) are chosen to model WMNs of different dimensions, using hop counts to measure distances. For example, we vary the distance between the gateway and an arbitrary MR, i.e., \(\alpha\) from 10 to 30 to simulate WMNs of different sizes. All costs presented below are normalized...
with respect to $\tau = 1$.

### 6.3.1 Performance Evaluation of APPCCM

![Graph](image)

**Figure 6.6:** Cost vs. $K/L$ (QUR = 0.5; QMR = 8.0).

![Graph](image)

**Figure 6.7:** Cost vs. $K/L$ (QUR = 2.0; QMR = 8.0).

Fig. 6.6 and Fig. 6.7 plot $C_{DPM}$ and $C_{CCM}$ as a function of $K/L$, under two different values of QUR. As both figures illustrate, for either mode, there exists an optimal threshold $K/L$ that minimizes the total communication cost incurred per time unit, given a set of parameters characterizing the mobility and data query/update characteristics (with respect to the accessed data object) of the MC. DPM performs consistently better than CCM, when QUR = 0.5, whereas CCM is superior to DPM when QUR = 2. Intuitively, DPM is expected to perform better than CCM under circumstances when the access rate is lower than the update rate, because DPM reduces invalidation costs at the expense of increased query costs. In contrast, CCM is expected to be
superior to DPM under other circumstances when the access rate is higher than the update rate, because CCM incurs smaller query costs than DPM under the same cache hit ratio, at the expense of increased invalidation costs.

The effect of tradeoff on the performance of DPM and CCM is clearly illustrated by Fig. 6.8, which compares $C_{DPM}$ and $C_{CCM}$, as a function of QUR. As can be seen in the figure, initially when QUR is small, DPM performs better than CCM. As QUR increases, the performance gap between DPM and CCM decreases, and there exists a crossover point of QUR beyond which CCM becomes superior to DPM.

Fig. 6.9 shows the optimal threshold $K_{optimal}$ in DPM as a function of QUR. As shown in both figures, the optimal threshold $K_{optimal}$ in DPM decreases with increasing QUR. This is because as QUR increases, the contribution of the query cost to the total communication cost becomes more
significant, and therefore, a smaller $K_{optimal}$ is favored to reduce the query cost and consequently minimize the total communication cost.

![Figure 6.10: Cost vs. QMR.](image)

Fig. 6.10 compares $C_{DPM}$ and $C_{CCM}$ as a function of QMR. We observe that this figure shows the same trend as in Fig. 6.8. There exists a crossover point of QMR beyond which CCM outperforms DPM. This is because with a fixed mobility rate and fixed update rates ($\mu$, $\delta$, and $\eta$), the access rate increases as QMR increases, and consequently QUR increases, resulting in the same trend as in Fig. 6.8.

![Figure 6.11: Cost vs. $P_{active}$ under two different values of QUR.](image)

Fig. 6.11 compares $C_{DPM}$ and $C_{CCM}$ as a function of $P_{active}$, under two different values of QUR. As the figure shows, Both $C_{DPM}$ and $C_{CCM}$ increase monotonically with increasing $P_{active}$. This is because the volume of activities of an MC and consequently the communication cost it incurs increase as the MC spends increasingly more time in active mode than in idle mode. The figure
also shows that DPM outperforms CCM when QUR = 0.5, whereas CCM is superior to DPM when QUR = 2.0. These results are well correlated with the trends exhibited in Fig. 6.6 and Fig. 6.7.

Fig. 6.12: Cost vs. QUR under different combinations of $\alpha$ and $\beta$.

Fig. 6.12 investigates the sensitivity of performance evaluation to $\alpha$ and $\beta$. Specifically, Fig. 6.12 compares $C_{DPM}$ and $C_{CCM}$ as a function of QUR, under different combinations of $\alpha$ and $\beta$. Comparing the figure with Fig. 6.8, it can be observed that both figures exhibit very similar trends, regardless of the values of $\alpha$ and $\beta$. Here we again observe that there exists a crossover point of QUR beyond which CCM outperforms DPM, and the performance gain of CCM over DPM increases with increasing QUR.

### 6.3.2 Comparative Performance Study

In this section, we compare APPCCM with two non-adaptive cache consistency management schemes for supporting mobile data access. The first non-adaptive scheme always caches a data object at the MC directly (we call it caching-at-the-MC scheme). Therefore, it is essentially equivalent to existing asynchronous stateful-based schemes [58,68] augmented with optimal per-user pointer forwarding for integrated cache consistency and mobility management such that MC can perform data access, data caching and cache consistency management while roaming and changing their locations in a WMN. This baseline scheme is selected in order to show the benefit of caching data objects adaptively, depending on QUR and QMR. The second non-adaptive scheme always caches a data object at the MC’s current serving MR (we call it caching-at-the-MR scheme). Therefore, a data object cached by an MC at its current serving MR will be migrated to the new serving MR every time the MC moves and changes its location. This baseline scheme is chosen to demonstrate the benefit of caching data objects adaptively as well as dynamically determining the optimal
threshold $K$.

Fig. 6.13 and Fig. 6.14 compare the total communication cost between APPCCM and the two non-adaptive schemes, as a function of QUR and QMR, respectively. We see that APPCCM outperforms both non-adaptive schemes. More specifically, APPCCM performs significantly better than the caching-at-the-MC scheme initially when QUR/QMR is small. As QUR/QMR increases, the performance gap narrows, and at some point, APPCCM degenerates to the caching-at-the-MC scheme, or equivalently, CCM. This is because APPCCM adaptively uses CCM to cache a data object when QUR/QMR is relatively large. Therefore, the advantage of APPCCM over the caching-at-the-MC scheme is most pronounced when QUR/QMR is relatively small.

Below we explain the reason that APPCCM is always superior to the caching-at-the-MR scheme. When QUR/QMR is small, APPCCM adaptively uses DPM to cache a data object. Since APPCCM
dynamically determines $K_{optimal}$ under which the total communication cost is minimized, APPCCM performs significantly better than the caching-at-the-MR scheme, when QUR/QMR is small. As QUR/QMR increases, APPCCM adaptively switches to CCM, because CCM is superior to DPM when QUR/QMR is large. Therefore, APPCCM is also superior to the caching-at-the-MR scheme when QUR/QMR is large. It is worth noting that the total communication cost is on a per time unit, per MC basis, so even a small performance gain of 5-10% will be significant over time. These results demonstrate the advantage of APPCCM over non-adaptive schemes due to its ability to choose either DPM or CCM that can best balance the query cost and the invalidation cost and thus minimize the overall cost.

6.3.3 Simulation Validation

We conduct extensive simulation to validate the analytical results obtained above, using a discrete simulation language called Simulation Model Programming Language (SMPL) [107]. In this simulation system, all operations in APPCCM are associated with discrete events, each with a state-dependent operation cost. For example, query processing operations, cache invalidation operations, data migration operations, and location update operations, are associated with events. Events are scheduled and executed in FIFO order, according to the algorithm description presented in Section 6.1. The average total communication cost incurred per time unit is evaluated and reported. To ensure the statistical significance of simulation results, we use a batch mean analysis technique. Each simulation batch consists of a large number of runs and therefore a large number of observations for computing one batch average. The simulation runs for a minimum of 10 batches, and stops until the mean of the batch means collected is within 5% from the true mean with a confidence level of 95%. In the simulation study we use the same set of parameter values as those listed in Table 6.5.

Fig. 6.15 and Fig. 6.16 show the simulation results of $C_{DPM}$ and $C_{CCM}$ as a function of $K/L$, under two different values of QUR. Comparing both figures with Fig. 6.6 and Fig. 6.7, we observe that the simulation results are well correlated with the analytical results, as demonstrated by the trends exhibited in the figures. This justifies that the analytical results are valid and there exists optimal thresholds $K_{optimal}$ and $L_{optimal}$, respectively, under which $C_{DPM}$ and $C_{CCM}$ are minimized.

Fig. 6.17 illustrates the simulation results of $C_{DPM}$ and $C_{CCM}$ as a function of QUR. Again, the simulation results show excellent correlation with the analytical results presented in Fig. 6.8.
Figure 6.15: Simulation validation: cost vs. $K/L$ (QUR = 0.5).

Figure 6.16: Simulation validation: cost vs. $K/L$ (QUR = 2.0).

Figure 6.17: Simulation validation: cost vs. QUR.
Similarly, the simulation results shown in Fig. 6.18 and Fig. 6.19 for the performance comparison between APPCCM and the two non-adaptive schemes also correlate well with the analytical results shown in Fig. 6.13 and Fig. 6.14.

6.4 Summary

In this chapter, we proposed an adaptive per-user per-object cache consistency management scheme for mobile data access in WMNs, namely APPCCM, with the objective to improve data access performance as well as to mitigate the performance bottleneck at the gateways in WMNs. APPCCM supports strong data consistency semantics through integrated cache consistency and mobility management. APPCCM is adaptive, per-user and per-object, as one of two caching modes provided by APPCCM, namely DPM and CCM, is adaptively selected based on the MC’s mobility and
data query/update characteristics as well as operational and networking conditions of the WMN. We demonstrated via model-based analysis that APPCCM outperforms two non-adaptive cache consistency management schemes that always cache a data object at the MC or at the MC’s current serving MR. The advantage of APPCCM is due to effective exploitation of the tradeoff between the query cost and invalidation cost realized by adaptively selecting the best cache consistency management strategy out of DPM and CCM.

There is a variety of potential applications for which APPCCM is applicable. For example, digital news and magazine applications running on smartphones and mobile tablet computers such as iPhone and iPad, are typical applications demanding mobile data access. It is beneficial to use APPCCM for such applications in a WMN environment to provide better response time, reduce the Internet traffic flow billed to the users, and minimize the total communication cost incurred for maximizing the network throughput.
Chapter 7

Integrated Mobility and Service Management for Mobile Multicast

In this chapter\(^1\), we propose and analyze an algorithm for supporting efficient mobile multicast in WMNs, called Dynamic Agent-based Hierarchical Multicast (DAHM), which supports potentially highly mobile users and dynamic multicast group membership. DAHM dynamically selects Multicast Agents (MAs) \([72]\) running on MRs for integrated mobility and multicast service management, and combines backbone multicast routing and local unicast routing into an integrated algorithm for efficient multicast packet delivery.

We evaluate and analyze the performance of DAHM via both analytical modeling and simulation validation. We also carry out a comparative performance study to compare DAHM with two baseline multicast algorithms, namely, Regional-Registration based Multicast (RRM) and Dynamic Tree-based Multicast (DTM). RRM is based on a hierarchical tree structure consisting of pure unicast paths, whereas DTM is based on a shortest-path multicast tree structure with dynamic updates upon member movement and group membership changes. Through the comparative performance study, we show that DAHM is superior to both RRM and DTM.

The remainder of this chapter is organized as follows. Section 7.1 gives a detailed introduction to DAHM. In Section 7.2 we develop an analytical model for analyzing the performance of DAHM. Detailed performance evaluation and a comparative performance study are given in Section 7.3, with both analytical results and simulation validation presented. The chapter is summarized in Section 7.4.

\(^1\)Part of this chapter is submitted as a journal paper to IEEE Transactions on Network and Service Management.
7.1 Dynamic Agent-based Hierarchical Multicast

7.1.1 Overview

DAHM supports efficient multicast packet delivery to potentially highly mobile multicast group members in a WMN, with the objective to minimize the overall network cost incurred by packet delivery, multicast tree maintenance, and mobility management. DAHM is a dynamic two-level hierarchical multicast algorithm featuring an integrated design that combines backbone multicast routing and local unicast routing. At the upper level of the hierarchy is the multicast backbone based on a shortest-path tree (SPT) rooted at the source whose tree nodes are MRs and whose leaves serve as MAs. The SPT is updated whenever an MA joins or leaves due to user mobility and group membership changes. Multicast packets are first disseminated from the source to all the MAs via multicast routing through the SPT, and then delivered from the MA to multicast group members individually via local unicast routing. The reason why unicast routing is used at the lower level rather than multicast routing as in [77,78] is twofold:

- The optimal service region size of an MA that minimizes the overall communication cost, i.e., the optimal threshold $H_{optimal}$ for the number of hops a multicast group member can be away from its MA, can be quite diverse for different group members depending on their mobility and service characteristics, as supported by the analytical and simulation results presented in Section 7.3. Therefore, group members associated with the same MA can have very diverse hop distances to the MA, making the wireless broadcast advantage no longer valid. Thus, using broadcast routing at the lower level can adversely affect the communication cost, because the overhead of multicast routing can be considerably high especially when a small number of receivers (group members associated with the same MA) are dispersed in a large service area around the sender (the MA).

- Using unicast routing eliminates the need for multicast tree maintenance at the lower level and simplifies mobility management. Suppose that multicast routing is used at the lower level, the need for mobility management as well as multicast tree maintenance would be frequent because multicast group members may have high mobility. Specifically, when a multicast group member moves to a new serving MR, the new serving MR needs to be subscribed to and the old serving MR needs to be unsubscribed from the multicast tree rooted at the MA, thus incurring two tree maintenance operations. When the member moves out of the service region of its current MA and switches to a new MA, not only changes to the multicast trees of both
the old and new MA need to be handled by the corresponding tree maintenance operations, group membership changes also need to be processed. If unicast routing is employed at the lower level, the overhead of multicast tree maintenance and multicast group membership management at the lower level would be completely eliminated. The saving can be significant, considering that group members can have high mobility and that the number of multicast groups at the lower level can be potentially large.

We use an SPT as the multicast backbone at the upper level as it is shown in [85] that an SPT is superior to a minimum cost tree (MCT) such as an approximate minimum Steiner tree (MST) in terms of packet delivery ratio, throughput, average end-to-end delay, and delay average jitter. Another advantage of an SPT over an MST is that the problem of constructing an MST is NP-complete. Lastly, considering SPT instead of sophisticated tree algorithms that strive for high throughput (e.g., [79,80,84]) allows us to focus on the design and analysis aspect of integrated mobility and multicast service management. Indeed, we could replace SPT with a more sophisticated algorithm and the design idea still applies. Here we note that our idea is generic as can be applied to other network services such as mobile data access [57].

An MA runs on an MR as a regional registration point for integrated mobility and multicast service management. Each multicast group member is registered with and serviced by an MA, from which it receives multicast packets via local unicast routing. The multicast group member
Figure 7.2: Message exchange sequence for a member join event.

also sends its updated location information, i.e., the address of its current serving MR, to the MA, whenever it moves and switches to a new serving MR. Each MA maintains a location database that stores the up-to-date location information of each multicast group member it currently services. Figure 7.1 illustrates the two-level hierarchical multicast structure employed by DAHM. In the remainder of this paper, we refer to a multicast group member simply as a member.

An MA and those members it currently services essentially form a local multicast group at the lower level of the hierarchy. Like the multicast backbone, a local multicast group is also dynamic due to user mobility and membership changes. Each MA covers a service region servicing all the members located within the region. The service region size of an MA is a key parameter controlling the tradeoff between the communication cost incurred at the upper level and that incurred at the lower level. There exists an optimal service region size that minimizes the overall communication cost. We model the optimal service region size as the optimal threshold for the number of hops a member can be away from its MA, denoted by $H_{\text{optimal}}$. This optimal threshold can be determined using the analytical model developed in Section 7.2. Below we let $H$ and $H_{\text{optimal}}$ denote the threshold and the optimal threshold, respectively.

### 7.1.2 Member Join and Leave

#### Member Join

An MC who intends to join a multicast group first selects a serving MR among all MRs within the wireless transmission range based on the wireless link quality, and sends a join request `JOIN_REQ` to the selected serving MR. If the new serving MR is not yet a leaf node of the multicast backbone, it needs to join the backbone multicast tree as a leaf node and becomes a new MA for the MC. The MR joins the backbone multicast tree by sending `JOIN_REQ` to the source. Upon receiving `JOIN_REQ`
Figure 7.3: Message exchange sequence for a member leave event (dashed lines mean conditional message exchanges).

from the MR, the source computes a shortest path to the MR and sends a join acknowledgment JOIN_ACK along the path back to it. The MR further forwards JOIN_ACK to the MC, confirming that it becomes a new member of the multicast group. By having the MA process member join requests locally, the signaling overhead of member join is significantly reduced. Figure 7.2 illustrates the procedure for a member join event. JOIN_REQ and JOIN_ACK also serve as an association request and an association acknowledgment, respectively.

Member Leave

When a member leaves a multicast group, it notifies its MA such that the MA can deregister it. After the member leaves, the MA may no longer service any member, therefore it needs to be removed from the backbone multicast tree. The procedure for a member leave event is illustrated in Figure 7.3. More specifically, the leaving member sends a leave request LEAVE_REQ to its MA, which responds with a leave acknowledgment LEAVE_ACK as a confirmation. If the MA needs to remove itself from the backbone multicast tree because it no longer services any member, it forwards LEAVE_REQ to the source. Upon receiving LEAVE_REQ from the MA, the source updates the backbone multicast tree and sends the MA a leave acknowledgment LEAVE_ACK in reply to the request.

In some cases, a member disconnects (either voluntarily or involuntarily) and therefore is not able to notify its MA. In DAHM, a member that disconnects is treated as a leaving member. The disconnection of a member can be detected by its MA when the MA tries to deliver multicast packets to the member. Once a member is detected to be disconnected, its MA deregister it and the MA needs to remove itself from the backbone multicast tree if it no longer services any member.
7.1.3 Mobility Management and Tree Maintenance

In DAHM, when a member moves and changes its serving MR, the following procedure is executed to handle the mobility management and multicast tree maintenance:

- When the member moves and switches to a new MR, it sends an association request `ASSO_REQ` to the new MR. The MR responds with an association acknowledgment `ASSO_ACK` in reply to the request, confirming that the association is completed and the MR becomes the new serving MR of the member.

- If the new serving MR is not an MA and is within the service region of the member’s MA, the member sends to its MA a location registration request `LOC_REG_REQ` containing the address of the new MR. The MA updates the member’s location information and sends a location registration acknowledgment `LOC_REG_ACK` back to the member. In this way, the MA always knows the up-to-date location information of members within its service region and is therefore able to deliver multicast packets to them individually through unicast routing.

- If the new serving MR is $H$ hops away from the member’s current MA, the threshold is reached
and the new MR needs to join the backbone multicast tree as a leaf node and becomes the new MA of the member. In this case, a join request $JOIN_{\text{REQ}}$ is sent to the source. Upon receiving $JOIN_{\text{REQ}}$ from the MR, the source computes a shortest path to the MR and sends a join acknowledgment $JOIN_{\text{ACK}}$ along the path back to it. Figure 7.4 shows the message exchange sequence in the case that the new MR is $H$ hops away from the member’s current MA.

- If the new MR is already an MA, the member switches to the new MA and starts receiving multicast packets from the new MA. Figure 7.5 shows the message exchange sequence in the case that the new MR is already an MA.

- After being associated with the new MA, the member sends a deassociation request $DEASSO_{\text{REQ}}$ to its old MA, which responds with a deassociation acknowledgment $DEASSO_{\text{ACK}}$.

- If the member’s old MA no longer services any member, it removes itself from the multicast tree by sending a leave request $LEAVE_{\text{REQ}}$ to the source. Upon receiving $LEAVE_{\text{REQ}}$ from the MA, the source updates the backbone multicast tree and sends the MA a leave acknowledgment $LEAVE_{\text{ACK}}$ as a confirmation.

### 7.1.4 Multicast Packet Delivery

In DAHM, multicast packets are delivered in a hierarchical manner from the multicast source to the multicast group members within a WMN. More specifically, multicast packet delivery in DAHM follows the following procedure:

1. If the source is a host in the Internet, it will first send multicast packets to the gateway, which is then responsible for distributing the packet to the MA. The gateway can be considered as a virtual source in this case.

2. For each multicast packet, the (virtual) source creates a new packet that encapsulates the multicast payload using a multicast address for the destination address field, and disseminates the new packet to the MA through the backbone multicast tree in multicast routing mode.

3. Upon receiving the packet, each MA decapsulates the packet and encapsulates the payload using the address of the serving MR of each member it services for the destination field, and forwards the new packet to the MR via unicast routing. The address of the serving MR of each member can be found in the MA’s location database.
Table 7.1: Parameters and notations used in performance modeling and analysis.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Physical meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\sigma$</td>
<td>Input</td>
<td>The average mobility rate of multicast group members</td>
</tr>
<tr>
<td>$\lambda_p$</td>
<td>Input</td>
<td>The multicast packet rate</td>
</tr>
<tr>
<td>SMR</td>
<td>Derived</td>
<td>Service to mobility ratio, defined as $\text{SMR} = \lambda_p / \sigma$</td>
</tr>
<tr>
<td>$\lambda$</td>
<td>Input</td>
<td>The rate of member join events</td>
</tr>
<tr>
<td>$\mu$</td>
<td>Input</td>
<td>The rate of member leave events</td>
</tr>
<tr>
<td>$M$</td>
<td>Input</td>
<td>The multicast group size</td>
</tr>
<tr>
<td>$n$</td>
<td>Input</td>
<td>The dimension of the WMN</td>
</tr>
<tr>
<td>$N$</td>
<td>Input</td>
<td>The number of MRs in the WMN</td>
</tr>
<tr>
<td>$\gamma$</td>
<td>Derived</td>
<td>The member density</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>Derived</td>
<td>The average unicast path length of the WMN</td>
</tr>
<tr>
<td>$\omega$</td>
<td>Derived</td>
<td>The arrival rate of a single member to an arbitrary MR</td>
</tr>
<tr>
<td>$P_{MA}$</td>
<td>Derived</td>
<td>The probability that an arbitrary MR is also an MA</td>
</tr>
<tr>
<td>$P_0$</td>
<td>Derived</td>
<td>The probability that an MR is not covering any member</td>
</tr>
<tr>
<td>$P_1$</td>
<td>Derived</td>
<td>The probability that an MR covers exactly one member</td>
</tr>
<tr>
<td>$P_{MA1}$</td>
<td>Derived</td>
<td>The probability that an MA services exactly one member</td>
</tr>
<tr>
<td>$N_{MA}$</td>
<td>Derived</td>
<td>The number of MAs</td>
</tr>
<tr>
<td>$T$</td>
<td>Derived</td>
<td>The multicast tree size in terms of the total number of tree nodes</td>
</tr>
<tr>
<td>$\kappa$</td>
<td>Input</td>
<td>The multicast scaling factor</td>
</tr>
<tr>
<td>$L$</td>
<td>Derived</td>
<td>The expected hop distance from the source to an MA</td>
</tr>
</tbody>
</table>

4. Each MR after receiving the multicast packet decapsulates the packet and delivers the packet to the designated member.

7.2 Performance Model

In this section, we develop a probability model based on stochastic Petri net (SPN) techniques [15] for evaluating the performance of DAHM. We choose SPN as the tool for performance modeling because: 1) an SPN model is a concise representation of the underlying Markov or semi-Markov chain that may have a large number of states; 2) an SPN model is capable of reasoning the behavior of a member, as it migrates among states in response to system events.

Table 7.1 lists the parameters and notations used in the following sections. The physical meaning of the mobility rate denoted by $\sigma$ is the average number of serving MR changes made by a multicast group member per time unit. The time unit used in this paper is second. If a group member moves and changes its serving MR once every 10 minutes, its mobility rate is $\frac{1}{600}$. The physical meanings of other parameters are clear from the context.
7.2.1 Stochastic Petri Net (SPN) Model

We assume that a WMN is structured as a two-dimensional $n \times n$ mesh with wraparound on the boundary such that each MR has exactly four neighbors, as illustrated in Figure 7.6. Each MR can communicate directly with any of its four neighbors that are within its communication range. A member can change randomly from its current serving MR to any of the MR’s four neighbors with equal probabilities of $\frac{1}{4}$. The total number of MRs in the network denoted by $N$ is simply given by $N = n^2$. The average unicast path length (hop count) denoted by $\alpha$ in this $n \times n$ mesh network model is given by [48]:

$$\alpha = \frac{2n}{3}$$  \hspace{1cm} (7.1)

Figure 7.7: The Markov chain modeling the process of arrival and departure of $M$ multicast group members to and from an MR.

We model the process of arrival and departure of $M$ multicast members to and from an MR using an $M/M/\infty/M$ queue. Figure 7.7 depicts the Markov chain for the $M/M/\infty/M$ queuing model, where $\omega$ means the arrival rate of a single member to an arbitrary MR, and is given by [73]:

$$\omega = \frac{\sigma}{n^2 - 1}$$  \hspace{1cm} (7.2)

Using the $M/M/\infty/M$ queuing model, the probability $P_0$ that an MR covers no members and the probability $P_1$ that an MR covers exactly one member can be derived as:

$$P_0 = (1 - \frac{1}{n^2})^M$$  \hspace{1cm} (7.3)
Figure 7.8: The SPN model for DAHM.

\[ P_1 = \frac{M}{n^2} (1 - \frac{1}{n^2})^{M-1} \] (7.4)

The dashed-line square within the mesh structure shown in Figure 7.6 illustrates the service region of an MA. Given that the threshold of the number of hops a member can be away from its MA is \( H \), the number of MRs within the service region of an MA on average is \( 2H^2 - 2H + 1 \), in the \( n \times n \) mesh network model. The probability denoted by \( P_{MA} \) that an arbitrary MR is an MA in DAHM is therefore given by:

\[ P_{MA} = \frac{1}{2H^2 - 2H + 1} \] (7.5)

Here we note that \( P_{MA} \) given above is only approximate because which MR is chosen as an MA for a multicast member depends on the user’s mobility. However, as validated by simulation reported in Section 7.3.3, this approximation does not affect the result accuracy.

An MA services exactly one member if all the MRs within its service region totally service exactly one member. Therefore, the probability denoted by \( P_{1MA} \) that an MA services exactly one member can be calculated as follows:

\[ P_{1MA} = \binom{2H^2 - 2H + 1}{1} \cdot P_{0}^{2H^2-2H} \cdot P_{1} \] (7.6)

At the upper level of the hierarchy, the number of MRs (including MAs) comprising the backbone multicast tree can be derived using the following method. First, the ratio of the total number of multicast links (among MRs) on the tree denoted by \( L_m \) over the average unicast path length of the network denoted by \( \alpha \) is given by a power-law [86,87] as follows:

\[ \frac{L_m}{\alpha} = R^\kappa \Rightarrow L_m = \alpha \cdot R^\kappa \] (7.7)

where \( \kappa \) is the *multicast scaling factor*, and is found to be close to 0.7 [86]. \( R \) denotes the number
Table 7.2: The meanings of places and transitions defined in the SPN model for DAHM.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Movement</td>
<td>mark(Movement)=1 means that the member moves and switches to a new serving MR</td>
</tr>
<tr>
<td>Hops</td>
<td>mark(Hops) returns the number of hops the member is away from its MA</td>
</tr>
<tr>
<td>Move</td>
<td>A timed transition modeling the movement of the member</td>
</tr>
<tr>
<td>MC2MA</td>
<td>A timed transition modeling the regional location registration event</td>
</tr>
<tr>
<td>Join</td>
<td>A timed transition modeling that the new serving MR joins the multicast tree as a leaf node and becomes a new MA</td>
</tr>
<tr>
<td>Reset</td>
<td>A timed transition modeling the event of registering with the new MR that is already an MA</td>
</tr>
</tbody>
</table>

of leaves on the multicast tree, i.e., the number of MA, and is calculated as:

\[ R = N_{MA} = P_{MA} \cdot N \]  \( (7.8) \)

Given \( L_m \), the total number of MRs (including the MAs) on the backbone multicast tree denoted by \( T \) is given as:

\[ T = L_m + 1 = \alpha \cdot (N_{MA})^\kappa + 1 \]  \( (7.9) \)

The expected hop distance \( L \) from the source to an MA is the average length of all paths from the source to the MAs, or, equivalently stated, it is equal to the average depth of all MAs (leaves) on the backbone multicast tree rooted at the source. Hence, assuming a perfectly balanced backbone multicast tree, \( L \) is calculated as follows:

\[ L = \log_d T \]  \( (7.10) \)

where \( d \) is the degree of an inner node (we use \( d = 4 \) because each inner node has four neighbors).

The optimal threshold for the number of hops a member can be away from its MA, denoted by \( H_{optimal} \), can be determined by using the SPN model. Figure 7.8 shows the SPN model for describing the behavior of a single group member. An SPN model consists of places, tokens, and transitions (for modeling events). Table 7.2 explains the meanings of places and transitions defined in the SPN model.

In Fig. 7.8 we put in numbers in parenthesis to label the SPN model sequence below. The SPN model is constructed as follows:

- The event of member movement is modeled by transition \( Move \), the rate of which is \( \sigma \). When a member moves and switches to a new serving MR, a token is put into place \( Movement \).
• The new MR may be either an ordinary MR or an MA. The SPN model distinguishes between these two cases using two immediate transitions $P1$ and $P2$ that are associated with probabilities $1 - P_{MA}$ and $P_{MA}$, respectively.

• In the first case that the new MR is not an MA, the member sends its current MA a LOC_REG_REQ message that contains the address of the new serving MR. Upon receiving the message, the MA updates the location information of the member stored in the location database, and acknowledges the location update by a LOC_REG_ACK message. The message exchange is modeled by transition $MC2MA$.

• After transition $MC2MA$ is fired, a token is put into place $Hops$. The number of tokens denoted by $mark(Hops)$ in place $Hops$ represents the number of hops the member is away from its MA.

• When the number of tokens in place $Hops$ reaches the threshold denoted by $H$, i.e., when $mark(Hops) = H$, transition $Join$ is fired, modeling that the new serving MR joins the backbone multicast tree as a leaf node and becomes the new MA of the member. The firing of transition $Join$ consumes all the tokens in place $Hops$.

• In the second case that the new MR is already an MA, the member registers with the new MA, and starts receiving multicast packets from the new MA. This is modeled by transition $Reset$, the firing of which consumes all the tokens in place $Hops$, meaning that the member is now directly serviced by the new MA and the hop counter is reset.

### 7.2.2 Performance Metrics

We use the total communication cost incurred per time unit as the metrics for performance evaluation and analysis, and the objective is to minimize this cost. We believe that minimizing this cost can have a positively impact on other metrics such as packet delivery ratio, throughput, average end-to-end delay, etc. The time unit is second in this paper. The average total communication cost incurred per member per time unit by DAHM, denoted by $C_{DAHM}$, includes the service cost for multicast packet delivery denoted by $\lambda_p \cdot C_s$, the signaling cost for mobility management denoted by $\sigma \cdot C_m$, the signaling cost for processing member join requests denoted by $\lambda \cdot C_j$, and the signaling cost for processing member leave requests denoted by $\mu \cdot C_l$. The equation for calculating $C_{DAHM}$
is therefore given as follows:

\[
C_{DAHM} = \lambda_p \cdot C_s + \sigma \cdot C_m + \lambda \cdot C_j + \mu \cdot C_l
\]  
\[\text{(7.11)}\]

\(C_s\), the service cost incurred per multicast group member per multicast packet delivery in DAHM, consists of two parts. The first part denoted by \(C_s^1\) is the total cost for disseminating the multicast packet from the source to all the MAs through the backbone multicast tree, namely \(T\), divided by the multicast group size \(M\). The number of wireless transmissions required to deliver a multicast packet from the source to the MA (i.e., \(C_s^1\)) equals \(T\) because each MR on the tree transmits the packet only once to all its downstream children \[88\]. The second part denoted by \(C_s^2\) is the average cost for delivering the multicast packet via unicast routing from an MA to a member it currently services. Since a member can be \(i\) hops away from its MA with probability \(P_i\) (0 \(\leq i \leq H - 1\), \(C_s^2\) is given by the probability-weighted average distance between the member and its MA. Therefore, \(C_s\) is the sum of the two parts:

\[
C_s = C_s^1 + C_s^2 = \frac{T}{M} + \sum_{i=0}^{i=H-1} P_i \cdot i
\]  
\[\text{(7.12)}\]

\(C_m\), the mobility management cost incurred per group member, depends on the event triggered by the movement of a member. More specifically, the mobility management cost is incurred when there is an MA join (\textit{Join} in the SPN model), MA reset (\textit{Reset} in the SPN model), or MA update (\textit{MC2MA} in the SPN model) event as follows:

- **MA join**: When the new serving MR of a member is \(H\) hops away from its MA after a movement, the new serving MR needs to join the backbone multicast tree as a leaf node and becomes the new MA of the member. In this event, the member completes the association with the new MR by sending an \texttt{ASSO\_REQ} message to it, which responds with an \texttt{ASSO\_ACK} message as an acknowledgment. The new MR joins the tree by sending a \texttt{JOIN\_REQ} message to the source, which computes a shortest path to the MR and sends a \texttt{JOIN\_ACK} message along the path back to it. With probability \(P_{\text{MA}}\), the member’s old MA no longer services any member, and it removes itself from the backbone multicast tree by sending a \texttt{LEAVE\_REQ} message to the source, which updates the tree and sends the MA a \texttt{LEAVE\_ACK} message.

- **MA reset**: When the new serving MR of the member is already an MA, the member switches to the new MA. In this event, the member completes the association with the new MA by sending an \texttt{ASSO\_REQ} message to the MA, which responds with an \texttt{ASSO\_ACK} message as an
acknowledgment. After being associated with the new MA, the member sends a \texttt{DEASSO\_REQ} message to its old MA, which responds with a \texttt{DEASSO\_ACK} message. With probability $P_{1}^{MA}$, the member’s old MA no longer services any member, and it removes itself from the backbone multicast tree by sending a \texttt{LEAVE\_REQ} message to the source, which updates the tree and sends the MA a \texttt{LEAVE\_ACK} message.

- **MA update**: When a member moves and changes its serving MR, the member sends to its MA a \texttt{LOC\_REG\_REQ} message containing the address of the new serving MR. The MA updates the location information of the member stored in the location database, and acknowledges the location update by a \texttt{LOC\_REG\_ACK} message.

Based on the discussion above, $C_m$ is given by:

$$
C_m = \begin{cases} 
2 + 2H + (1 + P_{1}^{MA}) \cdot 2L & \text{if “Join”} \\
2 + 2h + P_{1}^{MA} \cdot 2L & \text{if “Reset”} \\
2 + 2h & \text{if “MC2MA”}
\end{cases}
$$

(7.13)

where $h = \text{mark}(Hops)$ represents the distance between the member and its MA. We use $P_{1}^{MA}$ for the probability that the member is the only one currently serviced by its MA. Therefore, once the only member leaves, the MA will no longer service any member, and it should be removed from the backbone multicast tree.

$C_j$, the signaling cost per member join event, is computed as follows. An MC joins an existing multicast group by sending a \texttt{JOIN\_REQ} message to its newly selected serving MR. With probability $1 - P_{MA}$, the new serving MR is not yet a leaf node of the multicast backbone, and it needs to join the backbone multicast tree as a leaf node and becomes a new MA for the MC. The MR joins the multicast tree by sending \texttt{JOIN\_REQ} to the source, which responds with a \texttt{JOIN\_ACK} message as an acknowledgment. The MR further forwards \texttt{JOIN\_ACK} to the MC, confirming that it becomes a new member of the multicast group. Therefore, $C_j$ is calculated as:

$$
C_j = 2 + (1 - P_{MA}) \cdot 2L
$$

(7.14)

$C_l$, the signaling cost per member leave event, is computed as follows. The leaving member sends a \texttt{LEAVE\_REQ} message to its MA, which responds with a \texttt{LEAVE\_ACK} message as a confirmation. With probability $P_{1}^{MA}$, the MA needs to remove itself from the backbone multicast tree because it no longer services any member, and it forwards \texttt{LEAVE\_REQ} to the source, which updates the backbone multicast tree and sends the MA a \texttt{LEAVE\_ACK} message in reply to the request. Therefore, $C_l$ is calculated as:

$$
C_l = 2 + 2h + P_{1}^{MA} \cdot 2L
$$

(7.15)
where \( h = mark(Hops) \) represents the distance between the member and its MA.

The computational procedure outlined above can be easily implemented by first assigning a state-dependent cost as calculated from the equations above to each state of the underlying semi-Markov model and then computing \( C_{DAHM} \) by the state probability-weighted average cost, using the SPNP package [15].

### 7.3 Performance Analysis and Numerical Results

In this section, we evaluate the performance of the proposed hierarchical multicast algorithms, namely, DAHM, and the effect of various parameters on its performance. We also compare DAHM with two baseline multicast algorithms for WMNs, namely, RRM and DTM. Like DAHM, RRM is also a hierarchical multicast algorithm and it also employs MAs running on MRs for integrated mobility and multicast service management. However, the hierarchical tree structure in RRM is simply a union of pure unicast paths from the source to the group members. Therefore, RRM is a hierarchical unicast-based multicast algorithm. RRM is chosen as a baseline algorithm to show the benefit of maintaining a dynamic SPT-based multicast backbone and employing multicast routing at the higher level of the hierarchy. DTM transmits multicast packets through a dynamic shortest-path multicast tree whose leaves are MRs that directly service the members. The multicast tree in DTM is updated to maintain its structural properties every time a member moves and changes its serving MR. Therefore, DTM is essentially based on the existing multicast algorithm that relies on a shortest-path tree [85] augmented with the capability to perform dynamic tree updates for supporting member mobility and dynamic group membership. DTM is selected as a baseline algorithm to demonstrate the benefit of employing a dynamic hybrid multicast structure with MAs for integrated mobility and multicast service management as well as dynamically determining the optimal threshold \( H_{optimal} \).

To evaluate the effect of user mobility on the performance of the three algorithms, we introduce

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
<th>Typical value</th>
</tr>
</thead>
<tbody>
<tr>
<td>( M )</td>
<td>multicast group size</td>
<td>([10, 320])</td>
</tr>
<tr>
<td>( n )</td>
<td>network size</td>
<td>([5, 15])</td>
</tr>
<tr>
<td>( \lambda/\mu )</td>
<td>join to leave rate ratio of a member</td>
<td>1</td>
</tr>
<tr>
<td>SMR</td>
<td>service to mobility rate</td>
<td>([8, 6000])</td>
</tr>
</tbody>
</table>
a parameter called *service to mobility ratio* (SMR) defined as $\text{SMR} = \frac{\lambda}{\sigma}$. The physical meaning of SMR is the average number of multicast data packets transmitted from the source to a group member during the interval between two serving MR changes of the group member. SMR is an important parameter because it captures the service and mobility characteristics of a group member, both of which can have a significant impact on the operations of DAHM and on the overall network cost.

Table 7.3 lists the parameters and their values used in performance evaluation. These values are selected to demonstrate diversely sized multicast groups consisting of mobile members characterized by a broad range of SMR. The member join and leave rates are chosen to allow dynamically changing group membership, while maintaining a stable multicast group size. The range of $n$ is selected to model WMNs of reasonably diverse sizes.

### 7.3.1 Performance Evaluation

Figure 7.9 plots $C_{\text{DAHM}}$ as a function of the threshold $H$, under different multicast group sizes. It can be seen in the figure that there exists an optimal threshold $H_{\text{optimal}}$ that minimizes $C_{\text{DAHM}}$ for each different $M$. Fig 7.10 further shows $C_{\text{DAHM}}$ as a function of the threshold $H$, under different $n \times n$ network sizes. Again, the optimal threshold $H_{\text{optimal}}$ exists for each different $n$.

These results demonstrate that the service region size of an MA is key to the performance of DAHM, and there exists an optimal service region size that optimizes the performance of DAHM. The optimal service region size exists because of the tradeoff between the communication cost incurred at the upper level and that incurred at the lower level.

Figure 7.11 plots $C_{\text{DAHM}}$ as a function of the *member density* denoted by $\gamma$, which is defined
Figure 7.10: Cost vs. $H$, under different network sizes in DAHM ($M = 50$).

Figure 7.11: Cost vs. $\gamma$ in DAHM.
as $\gamma = \frac{M}{N}$, i.e., the average number of members serviced by one MR. As the figure shows, $C_{DAHM}$ decreases monotonically with increasing $\gamma$. This illustrates that multicast efficiency improves as the member density increases because the cost is effectively amortized by the increasing member population. The improvement in multicast efficiency is particularly significant at the upper level because the number of nodes on the backbone multicast tree increases sublinearly with increasing MAs ($\kappa < 1.0$).

Figure 7.12 shows the optimal threshold $H_{optimal}$ as a function of $\gamma$. It can be seen in the figure that $H_{optimal}$ decreases as $\gamma$ increases, and it drops to 1 when $\gamma$ is reasonably large. The service cost $C_s$ for multicast packet delivery in DAHM and accordingly $C_{DAHM}$ decrease with decreasing $H_{optimal}$, because the average distance over which multicast packets are transmitted at the lower level decreases. Therefore, the result conforms to the trend exhibited in Figure 7.11.

7.3.2 Comparative Performance Study

In this section, we compare DAHM with RRM and DTM, in terms of the average total communication cost incurred per member per time unit. RRM is a hierarchical multicast algorithm based purely on unicast routing. It is worth emphasizing that because the total communication cost is a per member per time unit metric, even a small cost reduction of 5% to 10% will be significant over time and over the entire group of members.

For RRM (DTM), the average total communication cost incurred per member per time unit denoted by $C_{RRM}$ ($C_{DTM}$ respectively) consists of the service cost for multicast packet delivery denoted by $\lambda_p \cdot C_s^{RRM}$ ($\lambda_p \cdot C_s^{DTM}$ respectively), the signaling cost for mobility management and multicast tree maintenance denoted by $\sigma \cdot C_m^{RRM}$ ($\sigma \cdot C_m^{DTM}$ respectively), the signaling cost for
processing member join requests denoted by \( \lambda \cdot C_j^{RRM} \) (\( \lambda \cdot C_j^{DTM} \) respectively), and the signaling cost for processing member leave requests denoted by \( \mu \cdot C_l^{RRM} \) (\( \mu \cdot C_l^{DTM} \) respectively). The following equations calculate \( C_{RRM} \) and \( C_{DTM} \):

\[
C_{RRM} = \lambda \cdot C_{s}^{RRM} + \sigma \cdot C_{m}^{RRM} + \lambda \cdot C_{j}^{RRM} + \mu \cdot C_{l}^{RRM}
\]

\[
C_{DTM} = \lambda \cdot C_{s}^{DTM} + \sigma \cdot C_{m}^{DTM} + \lambda \cdot C_{j}^{DTM} + \mu \cdot C_{l}^{DTM}
\]

The service cost for multicast packet delivery in RRM denoted by \( C_s^{RRM} \) consists of the cost of forwarding the packet from the source to the MA and the cost of delivering the packet from the MA to the group members they service, both via unicast routing. Therefore, \( C_s^{RRM} \) is calculated as:

\[
C_s^{RRM} = \frac{1}{M} (N_{MA} \cdot L + M \cdot \sum_{i=0}^{H-1} P_i \cdot i)
\]

(7.17)

\( C_m^{RRM} \) depends on the event triggered by the movement of a multicast group member. The equation for calculating \( C_m^{RRM} \) is the same as that for calculating \( C_m \) in DAHM. Additionally, the equations for calculating \( C_j^{RRM} \) and \( C_l^{RRM} \) are also the same as those for calculating the same cost terms in DAHM, because DAHM and RRM share the same message sequences for multicast structure maintenance and member join and leave events.

The service cost per multicast packet delivery in DTM is equivalent to the number of nodes on the multicast tree because each MR on the tree only transmits the packet once to its downstream children. Therefore, the service cost incurred per member is:

\[
C_s^{DTM} = \frac{T_{DTM}}{M}
\]

(7.18)

where \( T_{DTM} \) denotes the number of tree nodes on the shortest-path multicast tree in DTM. \( T_{DTM} \) can be calculated according to the power-law [86,87] as:

\[
T_{DTM} = \alpha R^k + 1
\]

(7.19)

where \( R \) denotes the number of leaf nodes on the multicast tree in DTM, i.e., the number of MRs that service at least one member, which is simply \( R = (1 - P_0) \cdot N \).

The tree maintenance cost in DTM consists of the costs of MR association and deassociation, and possibly the costs of multicast tree updates, as calculated by the following equation:

\[
C_m^{DTM} = 4 + (P_0 + P_1) \cdot 2L
\]

(7.20)

In DTM, when a member joins a multicast group, it establishes the association with a serving MR. With probability \( P_0 \), the MR needs to join the multicast tree as a leaf node because it is
not already a node on the tree. When a member leaves a multicast group, its association with its current serving MR is canceled. With probability $P_1$, the member is the only one that the MR services, and the MR needs to remove itself from the multicast tree because it will no longer service any member. Therefore, $C_j^{DTM}$ and $C_i^{DTM}$ are calculated as follows:

$$C_j^{DTM} = 2 + P_0 \cdot 2L$$  \hspace{1cm} (7.21)$$

$$C_i^{DTM} = 2 + P_1 \cdot 2L$$  \hspace{1cm} (7.22)$$

Figure 7.13 compares the average total communication cost incurred per member per time unit by the three algorithms as a function of the multicast group size $M$. As can be seen in the figure, the cost decreases as $M$ increases for all three algorithms. The reason is that the member density increases as $M$ increases, given that $n$ is fixed. This observation leads to the generalized conclusion

Figure 7.13: Performance comparison: cost vs. $M$ ($n = 10$).

Figure 7.14: Performance comparison: cost vs. $n$ ($M = 50$).
that multicast efficiency improves as the member density increases. It can also be seen in the figure that DAHM is superior to both RRM and DTM.

Figure 7.14 compares the total communication cost incurred per member per time unit by the three algorithms as a function of the network size $n$. As can be seen in the figure, for all the three algorithms, the cost increases with increasing $n$. This is because the member density decreases as $n$ increases, given that $M$ is fixed. Therefore, this observation also generalizes to the conclusion that multicast efficiency improves as the member density increases. Again, DAHM shows significantly better performance than both RRM and DTM. It is worth emphasizing again that because the total communication cost is a per member per time unit metric, even a small cost reduction of 5% to 10% will be significant over time and over the entire group of members.

Figure 7.15 studies the effect of the mobility rate denoted by $\sigma$ on the performance of the three algorithms, under different member densities. As can be seen in the figure, as SMR increases, the costs decrease monotonically because the contribution of the signaling cost for mobility management and multicast tree maintenance to the total communication cost decreases accordingly. As the figures show, DAHM performs consistently better than RRM and DTM over a wide range of SMR and the member density. DAHM copes well with the impact of high user mobility compared with
RRM and DTM, due to its capability to dynamically select the optimal service region size of an MA (i.e. $H_{\text{optimal}}$) that minimizes the total communication cost. RRM outperforms DTM when the members are highly mobile and the member density is low. However, the advantage diminishes as the member density increases. When the members have high mobility, DTM incurs a substantially larger signaling cost for mobility management and multicast tree maintenance, compared with DAHM and RRM. This is because DTM performs mobility management and tree maintenance every time a member moves and changes its serving MR. Additionally, when the member density is low, i.e., when a small number of members are sparsely distributed within the network, the multicast tree in DTM has a relatively large number of non-leaf MRs, leading to a relatively large cost for multicast packet delivery.

### 7.3.3 Simulation Validation

Here we conduct simulation experiments to validate the numerical data obtained in the previous sections. We implement the simulation system using a discrete event simulation language called Simulation Model Programming Language (SMPL) [107]. In this simulation system, all operations in DAHM are represented by discrete events associated with costs. For example, location update operations, multicast packet deliveries, member join/leave operations, and multicast tree maintenance operations, are all discrete events. Events are scheduled and executed in event occurrence time order, according to the algorithm description presented in Section 7.1. The average total communication cost incurred per member per time unit is evaluated and the mean cost is calculated periodically with an interval of 30 minutes in simulation time. To ensure the statistical significance.
of simulation results, we use batch mean analysis (BMA) techniques [107]. Each simulation batch consists of a large number of runs and therefore a large number of observations for computing an average. The simulation runs for a minimum of 10 batches, and stops until the calculated mean cost is within 5% from the true mean with a confidence level of 95%.

Figure 7.16 shows the analytical results versus the simulation results for $C_{DAHM}$ as a function of $H$, under different multicast group sizes. As the figure illustrates, the simulation results show excellent correlations with the analytical results. This justifies that the analytical results are valid and there exists an optimal service region size of an MA, under which DAHM is optimized. Similarly, excellent correlations between the analytical results and simulation results can be seen in Figure 7.17, which illustrates the analytical results versus the simulation results for $C_{DAHM}$ as a function of $\gamma$.

Figure 7.18 and Figure 7.19 plot the analytical results versus the simulation results for the
performance comparison among the three algorithms, as a function of $M$ and $n$, respectively. Again, the analytical results are well correlated with the simulation results in both figures. The perfect correlation between the analytical results and simulation results shown above justifies that the analytical results obtained in the paper are valid.

7.3.4 Discussion

Based on the analytical and simulation results presented above, we can draw the conclusion that DAHM significantly outperforms both RRM and DTM in a broad spectrum of configurations. This is because DAHM combines backbone multicast routing and local unicast routing into an integrated algorithm, and dynamically determines the optimal service region size of MAs to optimize multicast packet delivery, multicast tree maintenance, and group membership management collectively. Compared with RRM, the packet delivery cost at the upper level of the hierarchy in DAHM is significantly reduced. Compared with DTM, in addition to the reduction of the multicast packet delivery cost, the signaling cost for multicast tree maintenance and membership management in DAHM is significantly reduced.

7.4 Summary

In this chapter, we proposed an efficient multicast algorithms for WMNs, namely, Dynamic Agent-based Hierarchical Multicast (DAHM), which supports member mobility and dynamic group membership during the lifetime of a multicast group. DAHM employs a dynamic two-level hierarchical multicast structure, consisting of an upper-level backbone multicast tree rooted at the gateway with multicast agents as leaves, and lower-level local multicast groups rooted at the multicast agents.
DAHM leverages and dynamically selects multicast agents running on mesh routers for integrated mobility and multicast service management.

Each multicast agent maintains a location database that stores the location information of multicast group members it currently services, and handles location registration and location update operations locally. Multicast agents also serve as the connecting points for multicast packet delivery between the gateway (the virtual multicast source) and multicast group members. Multicast agents are dynamically selected and added to or removed from the backbone multicast tree due to the mobility of multicast group members and dynamic group membership changes. The optimal service region size of a multicast agent that optimizes the performance of DAHM can be dynamically determined using the analytical method presented in this chapter. Based on the analytical and simulation results obtained through a comparative performance study, we showed that DAHM significantly outperforms two baseline multicast algorithms for WMNs, namely, RRM and DTM.

RRM is chosen as a baseline algorithm to show the benefit of maintaining a dynamic SPT-based multicast backbone and employing multicast routing at the higher level of the hierarchy.

DTM is selected as a baseline algorithm to demonstrate the benefit of employing a dynamic hybrid multicast structure with MAs for integrated mobility and multicast service management as well as dynamically determining the optimal threshold $H_{optimal}$. 
Chapter 8

Integrated Mobility and Service Management for Reliable and Secure Mobile Multicast

Chapter 7 proposed and analyzed DAHM, an algorithm for supporting efficient mobile multicast in WMNs based on the design concept of integrated mobility and service management for mobile multicast. In this chapter\(^1\), we investigate the problem of supporting secure and reliable mobile multicast services in WMNs, and propose an algorithm called Hierarchical Agent-based Secure and Reliable Multicast (HASRM). HASRM is an extension to DAHM with support for reliable and secure mobile multicast. We identify the following requirements that HASRM must fulfill:

- The algorithm must take security measures to ensure that only authenticated members in a multicast group have access to the multicast data at any time. Particularly the algorithm must guarantee *forward secrecy* and *backward secrecy*. Secure multicast has received intensive attention due to its importance to group communications. It is particularly critical in WMNs because of the openness of wireless communications.

- The algorithm must handle failures of unreliable wireless links efficiently to guarantee that all group members receive each multicast packet. This guarantee is necessary for applications that require reliable multicast services, e.g., electronic newspaper and magazine delivery, and multi-site business data distribution.

- The algorithm must handle user mobility efficiently and support *mobile multicast* such that group members can receive multicast data when they move and change their locations (in

\(^1\)Part of this chapter is submitted as a journal paper to Performance Evaluation.
User mobility support is critical in WMNs because MC, which is end-user mobile devices, can have frequent movement.

HASRM is a decentralized hierarchical algorithm [96], by which a single multicast group is divided into subgroups managed locally by entities called Multicast Agents (MAs). Rekeying of security keys and group membership management are largely localized within the service region of an MA. An MA is an MR that besides being a regular wireless mesh router, also acts as a point of regional registration for integrated mobility and multicast service management. A multicast group member at all time is associated with a single MA, but may change its MA from time to time based on its mobility and multicast service characteristics. On the other hand, an MA may service multiple members simultaneously. HASRM dynamically determines the optimal regional service size of an MA (i.e., the number of MRs covered by the MA) that minimizes the overall network cost, based on the group’s multicast service characteristics and group dynamics in terms of member mobility and membership changes. HASRM achieves cost minimization by balancing the tradeoff between the cost for reliable multicast data delivery vs. the signaling cost for security, group membership, and mobility management tasks.

We develop a mathematical model based on stochastic Petri nets [15] to analyze the performance of HASRM with simulation validation, focusing on the effect of key parameters on the performance of HASRM. We demonstrate that HASRM under optimal settings (which can be derived dynamically using the proposed analytical model) significantly outperforms traditional algorithms based on shortest-path multicast trees extended with user mobility, security, and reliability support. We also compare HASRM with a recently proposed protocol framework for secure group communication in WMNs, called Secure Group Overlay Multicast (SeGrOM) [95]. We choose SeGrOM because it is also a hierarchical decentralized multicast algorithm based on a two-tier multicast structure. Like HASRM, it handles member mobility and dynamic group membership with decentralized management. We show that HASRM is superior to SeGrOM in terms of the overall network cost incurred by multicast data delivery, security key management, mobility management, and group membership management.

The remainder of this chapter is organized as follows. Section 8.1 gives a detailed introduction to HASRM. In Section 8.2 we develop an analytical model for evaluating the performance of HASRM. Performance analysis and detailed comparison results are presented in Section 8.3 and Section 8.4. Section 8.5 presents simulation results. The chapter is summarized in Section 8.6.
8.1 Hierarchical Agent-based Secure and Reliable Multicast

Like DAHM, HASRM employs a two-level hierarchical multicast structure. At the upper level of the hierarchy is a backbone multicast tree, which is a source-rooted multicast tree connecting MRs that serve as MAs. The tree is updated when an MA joins or leaves due to user mobility and group membership changes. HASRM dynamically maintains a group of MRs serving as MAs for integrated user mobility and secure and reliable multicast service management. Each multicast group member is registered with and serviced by an MA from which it receives secure and reliable multicast service. The member also reports its updated location information to the MA, whenever it moves and changes its serving MR. Each MA maintains a table that stores the up-to-date location information (the address of the current serving MR) of each multicast group member it currently services.

An MA and those members it services essentially form a local multicast group at the lower level of the hierarchy. Each MA services a region covering a number of MRs. The regional service size of an MA is a key parameter controlling the tradeoff between the packet delivery cost and the signaling cost for various management tasks. There exists an optimal regional service size that minimizes the overall communication cost. We model the optimal regional service size by the optimal threshold for the number of hops a member can be away from its MA, denoted by \( H_{\text{optimal}} \). This optimal threshold can be determined using the analytical model developed in Section 8.2. Let \( H \) and \( H_{\text{optimal}} \) denote the threshold and the optimal threshold, respectively.

8.1.1 Security Key Management

Each multicast group member shares a pair-wise secret key \( K_u \) with its MA, which uses the key to securely deliver multicast packets to the member. \( K_u \) can be established using the Diffie-Hellman (DH) key exchange protocol [104]. Whenever the member changes its MA due to its mobility, a new \( K_u \) shared with the new MA is generated. The source and the group of MAs share a group key \( K_g \) for data encryption that is used to transmit multicast packets securely from the source to the MA through the backbone multicast tree. To guarantee forward secrecy and backward secrecy, \( K_g \) must be updated every time an MA joins or leaves the backbone multicast tree.

- **MA join**: When an MA joins the backbone multicast tree, the old group key \( K_g \) is discarded and a new key \( K'_g \) is generated to ensure forward secrecy. The source and MA currently in the multicast group generate \( K'_g \) in a distributed way by applying a one-way hash function \( h \)
to $K_g$, i.e., $K'_g = h(K_g)$ [105]. Because the newly joining MA does not possess $K_g$, it cannot generate $K'_g$ using this method. Instead, the source securely sends $K'_g$ to the MA using its public key.

- **MA leave:** When an MA leaves the backbone multicast tree because it no longer services any multicast group member, $K_g$ needs to be updated to offer backward secrecy. The source generates a new shared group key $K'_g$ using the key tree approach [106] and distributes the key utilizing PKI to all MAs excluding the one that is leaving via rekey messages [106].

Table 8.1 summarizes the various security keys used in HASRM. HASRM uses conventional symmetric encryption and the DH protocol for key exchange to avoid the overhead associated with public-key encryption at the expense of the communication cost for key exchange. Considering that the packet rate is typically much higher than the mobility rate, the saving in the encryption/decryption cost is significant.

Table 8.1: Security keys used in HASRM.

<table>
<thead>
<tr>
<th>Key</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>$K_g$</td>
<td>The group key shared by the source and the group of MAs</td>
</tr>
<tr>
<td>$K_m$</td>
<td>The public key of an MA</td>
</tr>
<tr>
<td>$K_u$</td>
<td>The pair-wise secret key shared between a multicast group member and its MA</td>
</tr>
</tbody>
</table>

8.1.2 Reliable Multicast Data Delivery

The procedure for multicast data delivery when no failures of wireless links present is straightforward. The source first encrypts the packet using the group key $K_g$, and disseminates the encrypted packet to the subgroups (MAs) through the backbone multicast tree. Each MA, upon receiving the encrypted packet, decrypts the packet using the group key $K_g$, and re-encrypts the packet using the pair-wise key $K_u$ with a group member it serves in its service region, and then sends the new encrypted packet to this group member. The group member finally decrypts the packet using the pair-wise key $K_u$ shared with its MA. Therefore, a complete multicast transmission from the source to a group member is a two-stage process and involves two pairs of encryption/decryption operations.

In the presence of failures of wireless links that cause multicast data packet losses, error recovery based on retransmissions is necessary to provide reliable multicast data delivery. Specifically,
HASRM uses NAK-based retransmissions \cite{103}, i.e., a group member sends a negative acknowledgment when it missed a multicast data packet. Multicast data packet losses can be detected using gaps in the sequence numbers, as illustrated in Fig. 8.1. The following error recovery procedure is executed in case of packet losses:

- At the lower level of the hierarchy, when a multicast group member detects a multicast data packet loss, it sends a negative acknowledgment to its MA. The MA retransmits the lost packet to the group member when it receives the acknowledgment. Fig. 8.1 illustrates NAK-based retransmissions at the lower level of the hierarchy. Therefore, multicast data packet losses occurring at the lower level of the hierarchy are handled locally within the service region of each MA. In addition, the MA discards a buffered data packet after a maximum period of time for late negative acknowledgments to arrive and it also aggregates acknowledgments to limit acknowledgment implosion, so as to save bandwidth.

- At the upper level of the hierarchy, when an MA detects a gap in the sequence numbers, i.e., a multicast data packet loss, it sends a negative acknowledgment to the source. The source needs to retransmit the lost packet to the MA upon receiving the acknowledgment. Two options are available for retransmissions \cite{102}:

  - \textit{Multicast}: the source retransmits the lost multicast packet by multicasting the packet
to all MAs, regardless of whether they have already successfully received the packet or not.

- *Unicast*: the source individually retransmits the multicast packet to each of the MA that experienced the loss. MAs that successfully received the original packet will not receive the retransmitted one.

Which option is better depends on the loss probability of wireless links. If the loss probability is high, such that the number of MAs experiencing losses are relatively large, the multicast option is better because the signaling overhead of individual unicast will be significant. On the other hand, if the loss probability is not significantly high, the unicast option is better. We will only consider retransmissions based on unicast at the upper level of the hierarchy.

The lower level always uses unicast routing for multicast delivery from an MA to the group members associated with it. The reason of using unicast routing at the lower level is two-fold: First, the optimal service region size of an MA can be quite diverse for different group members depending on their diverse mobility and service characteristics. The result is that the wireless broadcast advantage is no longer valid, given that group members associated with the same MA (typically not a large number) can be highly dispersed in a large service area around the MA. Consequently, the overhead of multicast routing can be considerably high. Thus, using broadcast routing at the lower level can adversely affect the communication cost. Secondly, using unicast routing eliminates the need for multicast tree maintenance at the lower level and simplifies mobility management. The need for mobility management as well as multicast tree maintenance would be frequent if multicast routing is used at the lower level, as multicast group members may have high mobility. If unicast routing is employed at the lower level, the overhead of multicast tree maintenance and multicast group membership management at the lower level would be completely eliminated. The saving can be significant, considering that group members can have high mobility and that the number of multicast groups at the lower level can be potentially large.

It is also worth emphasizing that source-based retransmissions are only used at the higher level when an MA detects a data packet loss. At the lower level, however, multicast data packet losses and retransmissions are handled locally within the service region of each MA. When a multicast member detects a data packet loss, it would send a negative acknowledgment to its MA. The MA would process the acknowledgment locally by retransmitting the lost packet to the member, without relaying it to the source. Considering that the number of multicast group members is much larger
than that of the MA, this approach can significantly reduce the burden on the source.

8.1.3 Dynamic Group Membership Management

Member Join

To join a multicast group, an MC first selects a serving MR among all MRs within the wireless transmission range, say, based on the wireless link quality, and sends a *join request* to the selected MR. If the MR is not yet a part of the backbone multicast tree, it joins the backbone multicast tree by sending a join request to the source. The source computes a shortest path to the MR and sends a join acknowledgment along the path back to it. Any intermediate MRs on the path that is not a node on the backbone multicast tree automatically becomes an on-tree node when it receives the join acknowledgment. The MR forwards the acknowledgment to the MC, confirming that it becomes a new member of the multicast group. When joining the multicast group, the MC executes the DH protocol with the new serving MR to generate a new $K_u$.

Member Leave

To leave a multicast group, a member has to notify its MA. After the member leaves, the MA may no longer service any multicast group member, and therefore it needs to be removed from the backbone multicast tree. The leaving member sends a leave request to its MA, which responds with a leave acknowledgment as a confirmation. If the MA needs to remove itself from the backbone multicast tree because it no longer services any group member, it forwards the leave request to the source, which then updates the backbone multicast tree and sends the MA an acknowledgment.

8.1.4 Mobility Management

In HASRM, when a member moves and changes its serving MR, the following procedure is executed to handle mobility and backbone multicast tree management:

- The member associates with the new serving MR by sending an association request to it. The MR responds with an association acknowledgment.

- If the new serving MR is not an MA and is within the service region of the member’s MA, the member reports the serving MR change to its MA by a location update message. If the new serving MR is already an MA, the member switches to the new MA and starts receiving multicast packets from the new MA.
• If the new serving MR is $H$ hops away from the member’s current MA, the threshold is reached and the new serving MR sends a join request to the source to join the backbone multicast tree and becomes the member’s new MA.

• The member executes the DH protocol to generate a new key $K_u$ when it associates with the MA.

• After being associated with the new MA, the member sends a deassociation request to its old MA, which responds with a deassociation acknowledgment. If the member’s old MA no longer services any member, it removes itself from the backbone multicast tree by sending a leave request to the source.

8.2 Performance Model

In this section, we develop a stochastic Petri net (SPN) model for analyzing the performance of HASRM. Table 8.2 lists the parameters and their physical meanings used in performance modeling and analysis. The physical meaning of the mobility rate denoted by $\sigma$ is the average number of serving MR changes made by a multicast group member per time unit. The time unit used is second. If a group member moves and changes its serving MR once every 10 minutes, its mobility rate is $\frac{1}{600}$. The physical meanings of other parameters are clear from the context.

![Figure 8.2: An $n \times n$ mesh network model.](image)

We assume that the WMN has a two-dimensional $n \times n$ structure with wraparound on the boundary such that each MR has exactly four neighbors, as illustrated in Fig. 8.2. A multicast group member can move from its serving MR randomly to any of its four neighbors with equal probabilities. The average unicast path length denoted by $\alpha$ in this $n \times n$ mesh network model is given as $\alpha = \frac{2n}{3}$. 
Table 8.2: Parameters and their physical meanings.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Physical meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>σ</td>
<td>Input</td>
<td>The mobility rate</td>
</tr>
<tr>
<td>λ_p</td>
<td>Input</td>
<td>The multicast packet rate</td>
</tr>
<tr>
<td>λ</td>
<td>Input</td>
<td>The rate of member join events</td>
</tr>
<tr>
<td>μ</td>
<td>Input</td>
<td>The rate of member leave events</td>
</tr>
<tr>
<td>M</td>
<td>Input</td>
<td>The multicast group size</td>
</tr>
<tr>
<td>n</td>
<td>Input</td>
<td>The dimension of the WMN</td>
</tr>
<tr>
<td>α</td>
<td>Derived</td>
<td>The average unicast path length of the WMN</td>
</tr>
<tr>
<td>ω</td>
<td>Derived</td>
<td>The arrival rate of a single member to an arbitrary MR</td>
</tr>
<tr>
<td>P_{MA}</td>
<td>Derived</td>
<td>The probability that an arbitrary MR is also an MA</td>
</tr>
<tr>
<td>P_0</td>
<td>Derived</td>
<td>The probability that an MR is not covering any member</td>
</tr>
<tr>
<td>P_1</td>
<td>Derived</td>
<td>The probability that an MR covers exactly one member</td>
</tr>
<tr>
<td>P_{MA}^1</td>
<td>Derived</td>
<td>The probability that an MA services exactly one member</td>
</tr>
<tr>
<td>N_{MA}</td>
<td>Derived</td>
<td>The number of MAs</td>
</tr>
<tr>
<td>T</td>
<td>Derived</td>
<td>The multicast tree size in terms of the total number of tree nodes</td>
</tr>
<tr>
<td>κ</td>
<td>Input</td>
<td>The multicast scaling factor</td>
</tr>
<tr>
<td>p</td>
<td>Input</td>
<td>The loss probability of wireless links</td>
</tr>
<tr>
<td>P_h</td>
<td>Derived</td>
<td>The probability that a multicast data packet is successfully delivered along the path to a multicast group member from its MA which is ( h ) hops away</td>
</tr>
<tr>
<td>E_h</td>
<td>Derived</td>
<td>The expected number of retransmissions needed for an MA to deliver a multicast packet to a group member which is ( h ) hops away</td>
</tr>
<tr>
<td>L</td>
<td>Derived</td>
<td>The expected hop distance from the source to an MA</td>
</tr>
<tr>
<td>P_L</td>
<td>Derived</td>
<td>The probability that a multicast data packet is successfully transmitted from the source to an MA which is ( L ) hops away</td>
</tr>
<tr>
<td>E_L</td>
<td>Derived</td>
<td>The expected number of retransmissions needed for the source to disseminate a multicast packet to an MA which is ( L ) hops away</td>
</tr>
</tbody>
</table>

Figure 8.3: The Markov chain modeling the process of arrival and departure of \( M \) multicast group members to and from an MR.
We model the process of arrival and departure of $M$ multicast group members to and from an MR using an $M/M/\infty/M$ queue. Fig. 8.3 depicts the Markov chain for the $M/M/\infty/M$ queuing model. The probability $P_0$ that an MR is not servicing any group member and the probability $P_1$ that an MR services one member can be derived using the queuing model as:

$$
P_0 = (1 - \frac{1}{n^2})^M, \quad P_1 = \frac{M}{n^2}(1 - \frac{1}{n^2})^{M-1} \tag{8.1}
$$

It can be shown that, given the distance threshold $H$, the number of MRs covered by the service region of an MA (an example is given by the dotted-line-bounded area in Fig. 8.2) on average is $2H^2 - 2H + 1$. The probability $P_{MA}$ that an arbitrary MR is an MA is therefore given by:

$$
P_{MA} = \frac{1}{2H^2 - 2H + 1} \tag{8.2}
$$

An MA services exactly one member if all the MRs within its service region service just one member. Therefore, the probability $P_{1MA}$ that an MA services exactly one member can be calculated as:

$$
P_{1MA} = \left(\frac{2H^2 - 2H + 1}{1}\right) \cdot P_0^{2H^2-2H} \cdot P_1 \tag{8.3}
$$

At the upper level of the hierarchy, the number of MRs comprising the backbone multicast tree can be derived using the following method. First, the ratio of the total number of multicast links (among MRs) on the tree denoted by $L_m$ over the average unicast path length of the network denoted by $\alpha$ is given by a power-law [86,87] as follows:

$$
\frac{L_m}{\alpha} = R^\kappa \Rightarrow L_m = \alpha \cdot R^\kappa \tag{8.4}
$$

where $\kappa$ is the *multicast scaling factor*, and is found to be close to 0.7 [86]. $R$ denotes the number of leaves on the multicast tree, i.e., the number of MAs, and is calculated as:

$$
R = N_{MA} = P_{MA} \cdot N \tag{8.5}
$$

Given $L_m$, the total number of MRs (including the MAs) on the backbone multicast tree denoted by $T$ is given as:

$$
T = L_m + 1 = \alpha \cdot (N_{MA})^\kappa + 1 \tag{8.6}
$$

A multicast data packet is successfully delivered along the path from the MA to the member only if no links on the path lose the packet. Thus, the probability $P_h$ that a multicast data packet is successfully delivered along the path to a multicast group member from its MA which is $h$ hops away is calculated as follows:

$$
P_h = (1 - p)^h \tag{8.7}
$$
where \( p \) is the per-hop wireless link loss probability. Given \( P_h \), the expected number of retransmissions needed for an MA to deliver a multicast packet to a group member which is \( h \) hops away, i.e., \( E_h \), can be calculated as follows:

\[
E_h = \frac{1}{P_h} = \frac{1}{(1-p)^h}
\]  

The expected hop distance \( L \) from the source to an MA is calculated as the average length of paths from the source to the MA, i.e., the average depth of the MA (leaves) on the backbone multicast tree. Given that the number of tree nodes is \( T \) and let \( d \) denote the degree of inner nodes, \( L \) can be calculated as follows, assuming a perfectly balanced backbone multicast tree:

\[
L = \log_d T
\]  

Given \( L \), the probability \( P_L \) that a multicast data packet is successfully transmitted from the source to an MA which is \( L \) hops away from the source is given by:

\[
P_L = (1-p)^L
\]  

The expected number of retransmissions \( E_L \) needed for the source to disseminate a multicast packet to an MA is given as follows:

\[
E_L = \frac{1}{P_L} = \frac{1}{(1-p)^L}
\]

Figure 8.4: The SPN model for HASRM.

Here we present the SPN model for analyzing the performance of HASRM and particularly for determining the optimal threshold \( H_{optimal} \). Fig. 8.4 shows the SPN model for describing the behavior of a single group member. An SPN model consists of places, tokens, and transitions (for modeling events). Table 8.3 explains the meanings of places and transitions defined in the
Table 8.3: The meanings of places and transitions defined in the SPN model for HASRM.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Movement</td>
<td>mark(Movement)=1 means that the member moves to a new MR</td>
</tr>
<tr>
<td>Hops</td>
<td>mark(Hops) returns the number of hops the member is away from its MA</td>
</tr>
<tr>
<td>Move</td>
<td>A timed transition modeling the movement of the member</td>
</tr>
<tr>
<td>MC2MA</td>
<td>A timed transition modeling the regional location registration event</td>
</tr>
<tr>
<td>Join</td>
<td>A timed transition modeling that the new serving MR joins the multicast tree and becomes a new MA</td>
</tr>
<tr>
<td>Reset</td>
<td>A timed transition modeling the event triggered when the new serving MR is already an MA</td>
</tr>
<tr>
<td>$H$</td>
<td>The threshold for the number of hops a member can be away from its MA</td>
</tr>
</tbody>
</table>

SPN model. Here $\text{mark}(P)$ is a function that returns the number of tokens in place $P$. A token in this model represents a location change, i.e., a change of the serving MR of the group member. The underlying model of our SPN model is a semi-Markov chain, which when solved, yields the probability that the group member is in a particular state. Below we explain how the SPN model is constructed.

- The event of member movement is modeled by transition $\text{Move}$, the transition rate of which is $\sigma$. When the member moves to and is associated with a new MR, a token is put into place $\text{Movement}$.

- The new MR may be either an ordinary MR or an MA. The SPN model distinguishes between these cases using two immediate transitions $P1$ and $P2$ associated with probabilities $1 - P_{MA}$ and $P_{MA}$, respectively.

- In the first case that the new MR is not an MA, the member reports its new location to its MA. The event is modeled by transition $\text{MC2MA}$.

- After transition $\text{MC2MA}$ is fired, a token is put into place $\text{Hops}$. The number of tokens denoted by $\text{mark(Hops)}$ in place $\text{Hops}$ represents the number of hops the member is away from its MA.

- When the number of tokens in place $\text{Hops}$ reaches $H$, i.e., $\text{mark(Hops)} = H$, transition $\text{Join}$ is fired, modeling that the new MR joins the backbone multicast tree and becomes the new
MA for the member. The firing of transition Join resets the number of hops away from its current MA to zero. This is modeled by consuming all the tokens in place Hops.

- In the second case that the new MR is an MA, the member registers with the new MA, and the new MA replaces its old MA. This is modeled by transition Reset, the firing of which consumes all the tokens in place Hops.

We use the total communication cost incurred per time unit as the metric for performance evaluation and analysis. The reason we use this cost as the performance metric is that it reflects the network traffic load and, consequently, directly impacts traditional performance measurements such as throughput and latency.

Using the above metric, the average total communication cost incurred per member per time unit by HASRM, denoted by $C_{HASRM}$, includes the cost for reliable multicast packet delivery, the cost for mobility management, the cost for security key management, and the cost for group membership management. The service cost for reliable multicast packet delivery ($C_s$) consists of two parts. The first part ($C_1^s$) is for the transmissions/retransmissions at the upper level from the source to the MA. The second part ($C_2^s$) is for the transmissions/retransmissions at the lower level from the MA to the group members.

Assuming that unicast is used for retransmissions, $C_1^s$ is the sum of the cost for the initial multicast from the source to all MAs ($T$) and the cost for retransmissions from the source to all MAs ($(E_L - 1) \cdot 2L \cdot N_{MA}$), divided by the multicast group size ($M$). The cost for the initial multicast equals $T$ because each MA on the tree transmits the packet once to each of its downstream nodes [88]. The cost for retransmissions is $(E_L - 1) \cdot 2L \cdot N_{MA}$ to account for transmitting both the acknowledgments and retransmitted data packet along the paths between the source and the MA. Therefore, $C_1^s$ is calculated as follows:

$$C_1^s = \frac{1}{M} [T + (E_L - 1) \cdot 2L \cdot N_{MA}] \quad (8.12)$$

$C_2^s$ is given by the distance-probability-weighted costs for the transmissions/retransmissions from an MA to a group member, as follows:

$$C_2^s = \sum_{h=0}^{h=H-1} P_h \cdot h \cdot E_h \quad (8.13)$$

The cost for mobility management ($C_m$) depends on the transition event in the SPN model, i.e., Join, Reset, or $MC2MA$, triggered by the movement of a member. More specifically, $C_m$ is given
by:

\[
C_m = \begin{cases} 
2H + (1 + P_{MA}^1) \cdot 2L & \text{if Join} \\
2h + P_{MA}^1 \cdot 2L & \text{if Reset} \\
2h & \text{if MC2MA}
\end{cases}
\]

where \( h = \text{mark}(\text{Hops}) \) represents the distance between the member and its MA. When the Join event is triggered, the cost incurred includes three components: 1) the signaling cost for deassociation from the member’s current MA, 2) the signaling cost for the new MA to join the backbone multicast tree, and 3) the signaling cost for the current MA to be removed from the backbone multicast tree if it no longer services any group members. When the Reset event is triggered, the cost incurred includes the signaling cost for deassociation from the member’s current MA and the signaling cost for the MA to be removed from the backbone multicast tree if it no longer services any group members. Finally, the signaling cost incurred when the MC2MA event is triggered is for the MC to report a serving MR change to its MA, which is \( h \) hops away from the MC’s current serving MR.

The cost for security key management includes the cost for updating the group key \( K_g \) when an MA joins or leaves the backbone multicast tree (\( C_{Kg} \)), and the cost for a member to generate a new key \( K_u \) when it associates with a new MA (\( C_{Ku} \)). \( C_{Kg} \) is further divided into two parts, namely \( C_{Kg}^j \) and \( C_{Kg}^l \), for MA join and leave events, respectively. \( C_{Kg}^j \) is for the source to send the updated group key to the newly joining MA (existing MAs update the group key using a one-way hash function). \( C_{Kg}^l \) is for the source to send the updated group key to the MA excluding the one that left. \( C_{Kg}^j \) and \( C_{Kg}^l \) are therefore calculated as:

\[
C_{Kg}^j = 2L \\
C_{Kg}^l = (N_{MA} - 1) \cdot 2L
\]

\( C_{Ku} \) is the cost for a member to execute the DH protocol to generate a new key \( K_u \) when it changes its MA. Specific to the SPN model, \( C_{Ku} \) is incurred when transition Join or Reset is fired. The execution of the protocol involves a round-trip message exchange between the member and its new MA. Therefore \( C_{Ku} \) is calculated as:

\[
C_{Ku} = 2 \quad \text{if Join or Reset}
\]

The cost for group membership management consists of the cost for processing a member join event (\( C_j \)) and that for processing a member leave event (\( C_l \)). When a member joins the multicast group, the new serving MR of the member may need to be subscribed to the backbone multicast tree if it is not already an MA, the probability of which is \( 1 - P_{MA} \). In a member leave event, the
leaving member notifies its MA that is \( h \) hops away by a leave request, and if the MA no longer services any group member after the member leaves, it needs to be removed from the backbone multicast tree by forwarding the leave request to the source. Thus, \( C_j \) and \( C_l \) are calculated as:

\[
\begin{align*}
C_j &= (1 - P_{MA}) \cdot 2L \\
C_l &= 2h + P_{MA}^1 \cdot 2L 
\end{align*}
\] (8.17)

where \( h = \text{mark}(\text{Hops}) \) represents the distance between the member and its MA.

\( C_{HASRM} \) is the sum of each cost multiplied with the rate at which the associated operation occurs, i.e., \( C_{HASRM} \) is given by:

\[
C_{HASRM} = \lambda_p \cdot C_s + \sigma \cdot (C_m + C_{Kg}^j + C_{Ku}) + \lambda \cdot C_j + \mu \cdot (C_{Kg}^l + C_l)
\] (8.18)

### 8.3 Performance Evaluation

In this section, we evaluate the performance of HASRM and examine the effect of various parameters on its performance. To evaluate the effect of user mobility on the performance of the three algorithms, we introduce a parameter called *service to mobility ratio* (SMR) defined as \( SMR = \frac{\lambda_p}{\sigma} \). The physical meaning of SMR is the average number of multicast data packets transmitted from the source to a group member during the interval between two serving MRs changes of the group member. The time unit used is second. For example, if on average a group member changes its serving MR once every 10 minutes and the multicast packet rate is 10 (per second), its SMR is 6000. SMR is an important parameter because it captures the service and mobility characteristics of group members, both of which can have a significant impact on the operations of HASRM and on the overall network cost.

Table 8.4 lists the parameters and their values used in performance evaluation. These values are selected to demonstrate diversely sized multicast groups consisting of mobile members characterized
by a broad range of SMR. The member join and leave rates are chosen to allow dynamically changing group membership, while maintaining a stable multicast group size. The range of $n$ is selected to model a WMN of reasonably diverse sizes.

![Graph](image)

Figure 8.5: Cost vs. $H$, under different multicast group sizes in HASRM ($n = 10$).

![Graph](image)

Figure 8.6: Cost vs. $H$, under different network sizes in HASRM ($M = 50$).

Fig. 8.5 and Fig. 8.6 plot $C_{HASRM}$ as a function of the threshold $H$, under different multicast group sizes and network sizes, respectively. As the figures show, there exists an optimal threshold $H_{optimal}$ that minimizes $C_{HASRM}$ for each different $M$ and $n$. These results demonstrate that the regional service size of an MA is key to the performance of HASRM, and there exists an optimal regional service size that minimizes $C_{HASRM}$. It can also be observed that $C_{HASRM}$ decreases with increasing $M$ in Fig. 8.5, and that $C_{HASRM}$ decreases with decreasing $n$ in Fig. 8.6. These trends suggest that the multicast member population density, defined as $\gamma = \frac{M}{n^2}$, has a significant impact on the performance of HASRM, as illustrated in Fig. 8.7.
Fig. 8.7 shows $C_{HASRM}$ as a function of the multicast member population density $\gamma$. As can be seen in the figure, $C_{HASRM}$ decreases as $\gamma$ increases. This illustrates that multicast efficiency improves as the member population density increases because the cost is effectively amortized by the increasing member population.

Fig. 8.8 illustrates $H_{optimal}$ as a function of $\gamma$. We observe that $H_{optimal}$ decreases as $\gamma$ increases, and drops to 1 when $\gamma$ is reasonably large. Decreasing $H_{optimal}$ as $\gamma$ increases keeps $C_{HASRM}$ minimized. This is because as $\gamma$ increases, the cost incurred at the lower level of the multicast hierarchy will be dominating but its magnitude will reduce with decreasing $H_{optimal}$, thereby lowering $C_{HASRM}$. Fig. 8.8 demonstrates that HASRM can adapt to changes in member population density by dynamically determining $H_{optimal}$ that minimizes the total communication cost.
Fig. 8.9 plots $H_{optimal}$ as a function of $p$, the loss probability of wireless links, under different member population densities. As can be seen in the figure, the general trend is that $H_{optimal}$ decreases with increasing values of $p$. As $p$ increases, the service cost $C_s$ for multicast data delivery increases. Therefore, shorter paths for multicast data delivery are necessary to keep the total communication cost minimized, favoring smaller values of $H_{optimal}$ accordingly. This demonstrates one of the benefits of dynamically determining $H_{optimal}$ that minimizes $C_{HASRM}$, i.e., HASRM is dynamically adaptive to the quality of wireless links that may vary over time.

To further reveal the benefit of dynamically determining $H_{optimal}$ that minimizes $C_{HASRM}$, we compare HASRM with a special case of HASRM that uses static MAs with a fixed threshold $H$, say, $H = 4$, and we name this special case HASRM-S. HASRM-S represents an extension to a class of hierarchical reliable multicast algorithms that use statically placed proxies with fixed regional
service sizes for decentralized management. Specifically, the extension is for mobility and security support. Fig. 8.10 compares the total communication costs incurred by HASRM and HASRM-S respectively as a function of $\gamma$. As the figure shows, HASRM significantly outperforms HASRM-S, especially when $\gamma$ becomes large.

![Figure 8.10: HASRM vs. HASRM-S as a function of $\gamma$.](image)

Fig. 8.11 investigates the effect of SMR on the performance of HASRM and HASRM-S. We observe that HASRM significantly performs better than HASRM-S over a wide range of SMR values representing diverse user mobility and multicast service characteristics. HASRM outperforms HASRM-S in both cases because it can adapt to the changing member population density and movement frequency by dynamically determining the optimal MA regional service size ($H_{optimal}$) that keeps the total communication cost minimized.

### 8.4 Comparative Performance Study

We compare HASRM with traditional multicast algorithms based on a SPT [85] extended with user mobility, security, and reliability support (named the SPT algorithm). The SPT algorithm maintains an SPT rooted at the source with multicast group members as tree leaves for multicast data delivery. The multicast tree in the SPT algorithm is updated to maintain its structural properties every time a member moves and changes its serving MR. A group key $K_g$ is used for secure multicast data delivery from the source to the group members. Whenever a group member joins or leaves, the group key $K_g$ needs to be updated for all group members to ensure the forward and backward secrecy properties. NAK-based retransmissions are used to provide reliable multicast data delivery. The SPT algorithm is selected as a baseline algorithm to show the benefit.
of employing a dynamic hybrid multicast structure with MAs for integrated mobility and secure multicast service management as well as dynamically determining the optimal threshold $H_{optimal}$.

![Graph](image1.png)

**Figure 8.12:** HASRM vs. SPT as a function of $\gamma$.

Fig. 8.12 compares the total communication cost incurred as a function of $\gamma$, between HASRM and SPT. As expected, for both algorithms, the total communication cost decreases with increasing $\gamma$, because multicast efficiency improves as the member population density increases. As can be seen in the figure, HASRM is superior to SPT, particularly for moderate values of $\gamma$. It is worth emphasizing that because the total communication cost is a per member per time unit metric, even a small cost reduction of 5% to 10% will be significant over time and over the entire group of members.

![Graph](image2.png)

**Figure 8.13:** HASRM vs. SPT as a function of $p$.

Fig. 8.13 compares the total communication cost incurred as a function of $p$, the loss probability of wireless links, between HASRM and SPT. As expected, the total communication cost increases
with increasing $p$ for both algorithms, because the service cost for reliable multicast data delivery increases as $p$ increases. Again, HASRM performs significantly and consistently better than SPT. More importantly, it can be seen that HASRM copes much better than SPT with changing quality of wireless links.

We also perform a comparative performance study between HASRM and a recently proposed protocol framework for secure group communications in WMNs, called Secure Group Overlay Multicast (SeGrOM) [95]. SeGrOM is also a hierarchical decentralized multicast algorithm, and it handles member mobility and dynamic group membership with decentralized management. SeGrOM uses a two-tier multicast structure and two sub-protocols for multicast data delivery in the two tiers. The global data delivery protocol transmits multicast data via a secure overlay to head members, which are elected coordinators for local data delivery and group membership management. There is one coordinator for each subgroup of group members connected to the same MR. Whenever an coordinator leaves the MR to which it is connected, a new coordinator needs to be elected and subscribed to the secure overlay. If a member joins the multicast group via an MR without existing members, the member becomes a new coordinator and joins the secure overlay. The coordinator maintains a local data key shared among all members associated with the same MR for encrypting multicast packets. The local data key is refreshed (rekeyed) whenever a member associated with the same MR joins or leaves.

It can be seen that coordinators are similar to MAs except that coordinators are group members that are dynamic. Indeed, because each coordinator is always associated with an MR, the secure overlay in SeGrOM can be considered as consisting of a dynamic group of MRs. A difference is that the regional service size of an MA is dynamically determined by HASRM, whereas the service area of a coordinator is exactly the coverage area of an MR. To make the comparison between HASRM and SeGrOM on a fair basis, we will use a variant of HASRM that does not address reliability [97]. For similar reason, we compare the variant of HASRM with a variant of SeGrOM called SeGrOM-Group because like HASRM, this variant also uses a group key for data encryption and the secure overlay for group key distribution. It is worth noticing that SeGrOM does not address general mobility management, i.e., it does not consider mobility management as a general service that is necessary not only for mobile multicast, but also for other network services.

Fig. 8.14 compares the total communication cost incurred as a function of $\gamma$, between HASRM and SeGrOM. As can be seen in the figure, HASRM outperforms SeGrOM for a wide range of values of $\gamma$. SeGrOM is slightly better only when $\gamma$ is considerably large. Note that the total
communication cost incurred by HASRM also includes the signaling cost for mobility management, making HASRM applicable to any network services. It is worth emphasizing again that because the total communication cost is a per member per time unit metric, even a small cost reduction of 5% to 10% will be significant over time and over the entire group of members.

Fig. 8.15 further compares between HASRM and SeGrOM the total communication cost incurred as a function of SMR. As the figure shows, HASRM performs consistently better than SeGrOM for the investigated range of SMR, particularly when SMR is small, i.e., when the mobility rate is high. It shows that HASRM copes well with high group member mobility and is adaptive to the varying mobility rate. This is attributive to the design principle of integrated mobility and service management.
8.5 Simulation Validation

In this section, we conduct discrete-event simulations using a simulation language Simulation Model Programming Language (SMPL) \[107\] to validate the results obtained through the analytical model. To ensure the statistical significance of simulation results, we use a batch mean analysis technique. Each simulation batch consists of a large number of runs and therefore a large number of observations for computing one batch average. The simulation runs for a minimum of 10 batches, and stops until the mean of the batch means collected is within 5% from the true mean with a confidence level of 95%. In the simulation study we use the same set of parameter values as those listed in Table 8.4.

![Simulated cost vs. $H$](image)

Figure 8.16: Simulated cost vs. $H$, under different multicast group sizes in HASRM ($n = 10$).

Fig. 8.16 compares the analytical results vs. the simulation results for $C_{HASRM}$ as a function of $H$ under different multicast group sizes. The figure shows a perfect correlation between the analytical results and simulation results. Similarly, A perfect correlation between the analytical results and simulation results can be observed in Fig. 8.17, which compares the analytical results vs. the simulation results for $C_{HASRM}$ as a function of $\gamma$.

Fig. 8.18 compares the simulation results against the analytical results shown in Fig. 8.14. Again, the simulation results are perfectly correlated with the analytical results. These results demonstrate that the analytical model is valid and it accurately captures the operation of HASRM.

8.6 Summary

In this chapter, we proposed a hierarchical agent-based secure and reliable multicast (HASRM) algorithm for efficiently supporting secure and reliable mobile multicast in wireless mesh networks.
Figure 8.17: Simulated cost vs. $\gamma$ in HASRM.

Figure 8.18: Simulated HASRM vs. SeGrOM as a function of $\gamma$. 
Like DAHM, HASRM is also based on the design notion of integrated mobility and service management for mobile multicast. HASRM minimizes the overall communication cost incurred collectively by reliable multicast packet delivery, mobility management, security key management, and group membership management. HASRM achieves cost minimization by dynamically maintaining a group of MAs for integrated mobility and multicast service management and dynamically determining optimal regional service sizes of MAs as identified in the paper, when given a set of parameter values characterizing the networking environment, as well as the user mobility and multicast service behaviors. We demonstrated via a comparative performance study that HASRM significantly outperforms traditional algorithms based on shortest-path multicast trees extended with user mobility, security, and reliability support. We also showed that a variant of HASRM is superior to a recently proposed algorithm for secure group communications in WMNs.
Chapter 9

Integrated Mobility and Service Management in Mobile Ad Hoc Networks

In this chapter, we propose and analyze a scalable mobility management scheme for location-based routing in MANETs called Dual-region Mobility Management (DrMoM), which is based on the idea of employing local regions to complement existing location services in MANETs that assign home regions to mobile nodes and have mobile nodes in the home region of a mobile node serve as location servers for that node. DrMoM is based on the design notion of integrated mobility and service management for network cost minimization [70]. Specifically, unlike existing location services that define the home region size statically at design time, DrMoM dynamically determines the optimal home region size and local region size (defined by their respective radii denoted by \( R_h \) and \( R_l \)), which together minimize the overall network cost incurred by mobility management and data packet delivery. We develop a mathematical model for deriving the optimal values of the two key design parameters \( R_h \) and \( R_l \) and for calculating the overall network cost incurred by DrMoM, given system parameters characterizing the mobility and service characteristics of mobile nodes.

To demonstrate the benefit of our dual-region mobility management scheme, we compare DrMoM against a location-based routing protocol called SLURP [113] that handles mobility management using static home regions. We show that DrMoM under optimal settings outperforms SLURP in terms of the overall network cost incurred.

The chapter is organized as follows. A detailed introduction to DRMoM is presented in Section 9.1. Section 9.2 presents a mathematical model for analytically evaluating the performance of DrMoM. Section 9.3 performs a comprehensive performance evaluation, focusing on the effect of various parameters on the performance of DrMoM. The chapter is summarized in Section 9.4.
9.1 Dual-Region Mobility Management for Location-based Routing

9.1.1 Overview

We assume that mobile nodes are capable of tracking their locations, moving direction, and moving speed via a GPS module. We also assume that the density of mobile nodes is sufficiently high, so there is always at least one location server in each node’s home region.

In DrMoM, the coverage area of a MANET is statically partitioned into equally sized rectangular regions, as shown in Fig. 9.1. This global partitioning of the MANET coverage area is used as the basis for home region assignment. Specifically, each mobile node is permanently assigned a **home region**, whose center co-locates with the center of one of the rectangular regions, as illustrated by Fig. 9.1. The assignment is calculated by hashing the unique ID of the mobile node (e.g., its IP or MAC address) to the ID of one of the rectangular regions. We assume that every mobile node has knowledge about the global partitioning as well as the hash function such that it is able to locate the center of the home region of any node. All mobile nodes within the home region of a mobile node serve as **location servers** for that node. DrMoM varies the home region size dynamically based on the mobile node’s runtime mobility and service characteristics.

Besides the home region, each mobile node is also associated with a **local region**, and it exchanges location information with neighbors in the local region. Unlike the home region, which does not move, the local region moves with the mobile node. The home region keeps location summary information of the node, i.e., the coordinate of the center and radius of the node’s local region. Whenever the local region moves due to movement of the node, the location servers in the home region are updated with the location summary information. To locate the local region of a
destination node, the source node sends a location query to the destination node’s location servers.

The coordinates of the center of a home region is statically determined, whereas the radius is dynamically determined on a per-node basis, depending on the node’s mobility and service characteristics. The home region size, determined by its radius denoted by $R_h$, is a key factor balancing the tradeoff between the overhead for location queries/uploads and the robustness of the location service. Specifically, a larger home region covers more location servers on average and consequently increases the chance of a successful location query. However, a larger home region also leads to larger overhead for location queries and updates. Because $R_h$ is dynamic, the size of the home region is dynamic and not necessarily restricted by the size of the rectangular region. The local region size, determined by its radius denoted by $R_l$, is also a key parameter. Increasing the local region size increases the chance that a destination node is located using local location information, without querying the location servers. However, as the local region size increases, the cost of location inquiry packet delivery increases because of more hops to travel. The local region size also impacts on the rate of location updates to the home region, which is equal to the rate of local region boundary crossing.

Each mobile node maintains two location tables: the local region location table $LT_l$ that stores location information of nodes for which it is within their local regions, and the home region location table $LT_h$ that stores location information of nodes for which it serves as a location server. $LT_l$ is updated whenever the mobile node receives a local region location update, whereas $LT_h$ is updated whenever it receives a home region location update. An entry in $LT_l$ keeps the corresponding node’s “exact” location obtained from the most recent local region location update from that node. An entry in $LT_h$ stores the coordinates of the center and radius of the corresponding node’s local region obtained from the most recent home region location update from that node. A timestamp is associated with each entry in the tables to indicate its freshness and is copied into the header of data packets when the entry (for the destination) is used by the source node for data packet delivery. An entry expires after a time out period set depending on the granularity of the location information. Expired table entries are deleted periodically to make room for new entries.

To route data packets and control messages such as messages for location updates and queries, DrMoM uses geographical routing such as Most Forwarding with fixed Radius (MFR) [115] as illustrated in Fig. 9.2. For each hop, DrMoM selects the node from the one-hop neighbors of the current node that is closest to the destination (i.e., the node that makes the most progress towards the destination) as the next forwarding node. For example, in Fig. 9.2, node $Y$ is selected by the
source $S$ as the next forwarding node because it is closest to the destination $D$ among the neighbors of $S$. By selecting the next forwarding node this way, DrMoM guarantees that progress is made towards the destination for each hop, finally leading to the destination.

In the following sections, we present DrMoM in detail for location update, location query, data packet delivery, and home region maintenance operations. Table 9.1 lists the notations used.

### 9.1.2 Location Update

A mobile node uses a local region location update to notify neighbors within its local region of the coordinates of its current location as reported by the GPS module, and a home region location update to inform its location servers of the coordinates of the center and radius of its local region.

#### Local Region Location Update

Local region location updates follow a threshold-based approach. Specifically, a mobile node broadcasts a location update to its neighbors within its local region, when the distance between its current location and the location where the last update was triggered exceeds a threshold $\tau$. Each mobile node $S$ maintains a variable $S.loc_{last\_update}$ that records the location where the last local region location update was performed. Given a chosen value of $\tau$, the frequency of local region location updates depends on the mobility rate of $S$ [114]. In this paper, $\tau$ is set to be equal to the wireless transmission range such that the difference between the location of a mobile node kept by neighbors in its local region and its actual location is never larger than the wireless transmission range. Note that because the local region of $S$ is not restricted to its one-hop transmission range, a neighbor could potentially be multiple hops away.

The local region location update carries the following information about $S$: its node ID, current location ($S.curr\_location$), moving speed, and moving direction as reported by the GPS module. Whenever a local region location update is triggered, the radius of $S$’s local region, i.e., $R_l$, is
Table 9.1: The notations used in this paper.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S.id$</td>
<td>the node ID of $S$</td>
</tr>
<tr>
<td>$S.curr_location$</td>
<td>current location of $S$</td>
</tr>
<tr>
<td>$(S.hr_center, S.hr_radius)$</td>
<td>center and radius of $S$’s home region</td>
</tr>
<tr>
<td>$(S.lr_center, S.lr_radius)$</td>
<td>center and radius of $S$’s local region</td>
</tr>
<tr>
<td>$loc_info.ts$</td>
<td>timestamp of location information stored in $loc_info$</td>
</tr>
<tr>
<td>$S.loc_last_update$</td>
<td>location where the last local region location update was triggered</td>
</tr>
<tr>
<td>$local_loc_update$</td>
<td>local region location update message</td>
</tr>
<tr>
<td>$home_loc_update$</td>
<td>home region location update message</td>
</tr>
<tr>
<td>$local_loc_query$</td>
<td>local region location query message</td>
</tr>
<tr>
<td>$home_loc_query$</td>
<td>home region location query message</td>
</tr>
<tr>
<td>$NB(S)$</td>
<td>list of one-hop neighbors of $S$</td>
</tr>
<tr>
<td>$S.LT(D)$</td>
<td>result of lookup for $D$ in $S$’s location table (LT)</td>
</tr>
<tr>
<td>$dist(a, b)$</td>
<td>distance between location $a$ and location $b$</td>
</tr>
<tr>
<td>$m.Header_Loc_Info(D)$</td>
<td>location information of $D$ carried in the header of packet $m$</td>
</tr>
<tr>
<td>$S.Forward_Packet(m, NB(S))$</td>
<td>procedure for $S$ to forward a packet $m$ to the next hop towards the destination</td>
</tr>
<tr>
<td>$S.Broadcast(m, NB(S))$</td>
<td>procedure for $S$ to broadcast $m$ to its one-hop neighbors</td>
</tr>
<tr>
<td>$S.Send_Reply(D, m)$</td>
<td>procedure for $S$ to send the reply $m$ to $D$</td>
</tr>
<tr>
<td>$S.Update_Entry(LT(D), m)$</td>
<td>procedure for $S$ to update the entry for $D$ in $LT$ using location information carried by $m$</td>
</tr>
</tbody>
</table>
| $S.Update\_Header(S, loc\_info)$ | procedure for $S$ to update the data packet header using $loc\_info$ |}
| $S.Calculate\_Radius(selector)$ | procedure for $S$ to dynamically calculate the radius for the local region or home region |
re-calculated and the up-to-date value is also carried by the location update. This information is necessary for neighboring nodes to dynamically determine if they are within $S$’s local region, since the value of the radius may change over time, depending on $S$’s mobility rate and service characteristics. Each neighboring node receiving the location update determines if it is within $S$’s local region by comparing its distance to the current center of $S$’s local region against $R_l$. If its is within $S$’s local region, it updates its local $LT_l$, and rebroadcasts the location update. If a neighboring node determines that it is outside $S$’s local region, it simply drops the location update. Algorithm 2 describes the procedure for processing a local region location update.

**Algorithm 2:** Processing a local region location update.

```markdown
if dist($S$.loc_last_update, $S$.curr_location) ≥ $\tau$ then
    $S$.loc_last_update = $S$.curr_location;
    $S$.Calculate_Radius($S$.RL);
    $S$.Broadcast(local_loc_update, NB($S$));
    foreach neighboring node $i$ do
        /* if $i$ is within $S$’s local region */
        if dist($i$.curr_location, $S$.curr_location) < $S$.RL then
            $i$.Update_Entry($i$.LT($S$), local_loc_update);
            $i$.Broadcast(local_loc_update, NB($i$));
        else
            $i$.drops local_loc_update;
```

**Home Region Location Update**

A home region location update is triggered whenever a mobile node $S$ moves outside its current local region. Specifically, when $S$ detects that it has moved outside its current local region (by comparing its current location with the center of its local region), it calculates the radius $R_l$ for the new local region and sends a home region location update to the location servers in its home region. The location update carries the coordinates of the center and radius of $S$’s new local region. The radius of $S$’s home region, i.e., $R_h$, is also re-calculated based on $S$’s mobility and service characteristics every time a home region location update is triggered, and is also carried by the location update. This information is necessary for other nodes to dynamically determine and mark if they are within $S$’s home region and serve as location servers for $S$ by comparing their distances to the center of $S$’s home region with $R_h$.

The first location server within the home region receiving the location update broadcasts the update in the home region, as in [113]. Each subsequent node receiving the location update checks
if it is within $S$’s home region by comparing its location to the center of $S$’s home region (the node obtains this information by applying the hash function to $S$’s node ID) against the radius $R_h$ carried by the location update. If the node is within the home region, it updates its local $LT_h$ and rebroadcasts the location update. If the node determines that it is outside $S$’s home region, it simply drops the location update. To reduce bandwidth and energy consumption and network congestion, each location server only broadcasts the first received update after waiting for a random amount of time.

Given a chosen value of $R_l$, the frequency of home region location updates depends on $S$’s mobility rate. Algorithm 3 gives the procedure for processing a home region location update.

```
Algorithm 3: Processing a home region location update.

if dist(S.curr_location, S.lr_center) ≥ S.R_l then
    S.Calculate_Radius(S.R_l);
    S.lr_center = S.curr_location;
    S.Calculate_Radius(S.R_h);
    S.Forward_Packet(home_loc_update, NB(S));
    foreach intermediate node i do
        /* if i is the first location server of S receiving the update */
        if dist(i.curr_location, S.hr_center) < S.R_h then
            i.Update_Entry(i.LT_h(S), home_loc_update);
            i.Broadcast(home_loc_update, NB(i));
            /* Break out of the loop */
            Break;
        else
            i.Forward_Packet(home_loc_update, NB(i));
    endforeach node j receiving the broadcast location update do
        /* if j is within S’s home region */
        if dist(j.curr_location, S.hr_center) < S.R_h then
            j.Update_Entry(j.LT_h(S), home_loc_update);
            j.Broadcast(home_loc_update, NB(j));
        else
            j drops home_loc_update;
```

### 9.1.3 Location Query

A location query is a two-stage procedure: the first stage is a local region location query, optionally followed by a home region location query in the second stage if no location replies have been received in the first stage.
Local Region Location Query

In the first stage, the source node $S$ broadcasts a location query within its local region, hoping that some neighboring nodes have the up-to-date location information of the destination node $D$. In the meantime, $S$ starts a timer that expires after approximately the time for a round-trip transmission between $S$ and the furthest neighbor in its local region (to ensure that location replies if any are received before the timer expires). The expiration of the timer before any location reply is received indicates a failed local region location query. The local region location query carries the current coordinates of the center and $R_l$ of $S$’s local region for neighboring nodes to dynamically determine if they are within $S$’s local region.

Any neighboring node in $S$’s local region that finds valid location information of $D$ in either $LT_l$ or $LT_h$ sends a location reply to $S$. The location reply carries the timestamp of the table entry for $D$ indicating the freshness of the location information. Upon the expiration of the timer, $S$ collects all replies and uses the one with the most recent location information of $D$. To prevent a reply storm from happening, each neighbor waits for a random amount of time before sending out the reply [113]. The procedure for processing a local region location query is presented in Algorithm 4.

Algorithm 4: Processing a local region location query.

```plaintext
S.Broadcast(local_loc_query, NB(S));
foeacneighboring node i do
    /* if i is within S’s local region */
    if dist(i.curr_location, S.lr_center) < S.R_l then
        if i.LT_l(D) ≠ null OR i.LT_h(D) ≠ null then
            i.Send_Reply(S, local_loc_reply);
            i.Broadcast(local_loc_query, NB(i));
        else
            i drops local_loc_query;
    S picks local_loc_reply with the latest timestamp;
    S.Update_Table(S.LT_l(D), local_loc_reply);
```

Home Region Location Query

A home region location query is triggered in the second stage if the local region location query fails to locate $D$ in the first stage. Specifically, $S$ locates the home region of $D$ by applying the hash function to $D$’s node ID and sends a location query towards the center of the home region of $D$ using geographical routing. The location query will ultimately reach a location server within the home region of $D$, which retrieves the entry for $D$ in its $LT_h$, and sends a location reply back to $S$. 
Algorithm 5 describes the procedure for processing a home region location query.

<table>
<thead>
<tr>
<th>Algorithm 5: Processing a home region location query.</th>
</tr>
</thead>
</table>
| \[
S\text{.Forward\_Packet}(\text{home\_loc\_query}, \text{NB}(S));
\]
| \[
\textbf{foreach} \text{ intermediate node } i \text{ do}
\]
| \[
\text{if } i \text{ is a location server of } D \text{ then}
\]
| \[
i\text{.Send\_Reply}(S, \text{home\_loc\_reply});
\]
| \[
/* \text{Break out of the loop } */
\]
| \[
\text{Break};
\]
| \[
\text{else}
\]
| \[
i\text{.Forward\_Packet}(\text{home\_loc\_query}, \text{NB}(i));
\]
| \[
S\text{.Update\_Table}(S.LT_h(D), \text{home\_loc\_reply});
\]

9.1.4 Data Packet Delivery

Suppose the source node $S$ has a data packet $m$ to send to the destination node $D$. $S$ needs to locate $D$ first by looking up the location information of $D$ in its $LT_l$ and $LT_h$. Depending on the result of this table lookup, there could be three cases as follows:

- **Case 1**: A valid entry for $D$ exists in $LT_l$. $S$ uses Algorithm 6 for data packet delivery.

- **Case 2**: A valid entry for $D$ exists in $LT_h$. $S$ uses Algorithm 6 for data packet delivery.

- **Case 3**: No valid entry for $D$ can be found because the entry has expired or no entry for $D$ exists. In this case, $S$ initiates a location query before sending any data packets to $D$. Upon receiving the location reply, $S$ updates its location tables and uses Algorithm 6 for data packet delivery.

The procedure for processing data packet delivery given that a valid entry for $D$ exists in either $LT_l$ or $LT_h$ is presented in Algorithm 6. A potential optimization is that in addition to the data payload, $m$ also carries the up-to-date location information of the sender $S$ such that intermediate mobile nodes can update their location tables using such location information. Data packets are forwarded towards the destination using geographical routing.

9.1.5 Maintenance of Home Region

Because location servers within the home region of a mobile node are also mobiles, they may leave the home region and therefore no longer serve as location servers. Other nodes may also move
### Algorithm 6: Processing data packet delivery.

```plaintext
if \( S.LT_l(D) \neq \text{null} \) then
    \[ S.\text{Forward\_Packet}(m, NB(S)); \]
    \[ \text{foreach} \ \text{intermediate node } i \ \text{do} \]
    \[ \begin{align*}
    & \text{if } i.LT_l(D) \neq \text{null} \ \text{and } i.LT_l(D).ts \geq m.\text{Header\_Loc\_Info}(D).ts \text{ then} \\
    & \quad i.\text{Update\_Header}(m, i.LT_l(D)); \\
    & \quad i.\text{Forward\_Packet}(m, NB(i));
    \end{align*} \]
else if \( S.LT_h(D) \neq \text{null} \) then
    \[ S.\text{Forward\_Packet}(m, NB(S)); \]
    \[ \text{foreach} \ \text{intermediate node } i \ \text{do} \]
    \[ \begin{align*}
    & \text{if } i.LT_h(D) \neq \text{null} \ \text{and } i.LT_h(D).ts \geq m.\text{Header\_Loc\_Info}(D).ts \text{ then} \\
    & \quad i.\text{Update\_Header}(m, i.LT_h(D)); \\
    & \quad i.\text{Forward\_Packet}(m, NB(i));
    \end{align*} \]
```

into the home region and become new location servers for the node. A location server \( B \) for a node \( A \) can detect whether it is within \( A \)'s home region by periodically checking its distance to the center of \( A \)'s home region against the radius \( R_h \) of \( A \)'s home region (\( B \) knows \( R_h \) from home region location updates sent by \( A \)). If \( B \)'s distance to the center of \( A \)'s home region is larger than \( R_h \), \( B \) is no longer within \( A \)'s home region. After \( B \) leaves \( A \)'s home region, it will not receive home region location updates from \( A \), and it will ignore any home region location queries for \( A \)'s location information even though it may still receive it (because the querying node may not have the up-to-date information on the radius of \( A \)'s home region).

When a node \( C \) moves and enters into the home region of \( A \), \( C \) is notified of the current location information of \( A \) by existing location server nodes in the home region. DrMoM requires that each node serving as a location server in the home region of \( A \) periodically broadcast to its neighbors a message announcing its identity as a location server for \( A \). The message carries the node ID of \( A \), the coordinates of the center and \( R_h \) of \( A \)'s home region, and the current location information of \( A \). When \( C \) receives the message from one of \( A \)'s location servers, it checks if it is within \( A \)'s home region by comparing its distance to the center of \( A \)'s home region against the \( R_h \). If \( C \) detects that it is within \( A \)'s home region, it stores the location information of \( A \) in its \( LT_h \) and starts serving as a location server for \( A \).

### 9.1.6 Application of Optimal Setting

Given the optimal setting (i.e., the optimal \( R_l \) and \( R_h \)), DrMoM requires each mobile node to apply the optimal setting at runtime to make it effective. Specifically, a mobile node \( S \) broadcasts...
an update message that carries the optimal values of $R_l$ and $R_h$, its current location, and the coordinates of the center of its home region. Each mobile node upon receiving the update message determines if it is within the local region or home region of $S$ based on information carried in the update and its own location and takes the responsibility accordingly.

9.2 Mathematical Model for DrMoM

In this section, we present a mathematical model for calculating the parameterized overall communication cost incurred by DrMoM as a function of $R_l$ and $R_h$. We define the total communication cost incurred by DrMoM for mobility management and data packet delivery by the total number of wireless transmissions per time unit. It is worth emphasizing that because the total communication cost is a per time unit metric, a small amount of communication cost savings can be significant over time. Table 9.2 lists the notations used for model parameters.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Type</th>
<th>Physical meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n$</td>
<td>Input</td>
<td>Total number of mobile nodes in the MANET</td>
</tr>
<tr>
<td>$r$</td>
<td>Input</td>
<td>Wireless transmission range</td>
</tr>
<tr>
<td>$R_l$</td>
<td>Design</td>
<td>Radius of a local region</td>
</tr>
<tr>
<td>$R_h$</td>
<td>Design</td>
<td>Radius of a home region</td>
</tr>
<tr>
<td>$b(R)$</td>
<td>Derived</td>
<td>Broadcast cost in a region with radius $R$</td>
</tr>
<tr>
<td>$v$</td>
<td>Input</td>
<td>Moving speed (m/s) of a mobile node</td>
</tr>
<tr>
<td>$\sigma$</td>
<td>Derived</td>
<td>Crossing rate of local region boundaries of a mobile node</td>
</tr>
<tr>
<td>$\bar{d}$</td>
<td>Derived</td>
<td>Average distance between a mobile node and its home region</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>Derived</td>
<td>Average number of hops between a mobile node and its home region</td>
</tr>
<tr>
<td>$\gamma$</td>
<td>Derived</td>
<td>Node density (average number of nodes per unit area)</td>
</tr>
<tr>
<td>$\lambda_l$</td>
<td>Derived</td>
<td>Rate of local region location updates</td>
</tr>
<tr>
<td>$\lambda_h$</td>
<td>Derived</td>
<td>Rate of home region location updates</td>
</tr>
<tr>
<td>$\mu$</td>
<td>Derived</td>
<td>Rate of home region maintenance</td>
</tr>
<tr>
<td>$\phi$</td>
<td>Input</td>
<td>Data packet rate</td>
</tr>
<tr>
<td>$\zeta$</td>
<td>Input</td>
<td>Session rate</td>
</tr>
</tbody>
</table>

9.2.1 Assumptions

We make the following assumptions when building the mathematical model:

- We use the modified random way point mobility model [49] to simulate the movement of mobile nodes. Specifically, each node picks a random point and it moves towards that point
with velocity $v$ randomly chosen in the range $[v_{\text{min}}, v_{\text{max}}]$. $v_{\text{min}}$ is positive to avoid the problem of speed decay [49] as time progresses. Once the point is reached, the node chooses a new random point and moves towards the new point without a pause.

- We assume that the hash function used for home region assignment maps any mobile node uniformly to any rectangular region with equal probability.

- We assume a uniform traffic pattern, i.e., the probability of packet transmissions between any pair of nodes is the same.

- We assume that nodes are uniformly distributed in the geographic area of the MANET, i.e., the node density is homogeneous across the entire MANET area. We further assume that the geographic area of the MANET is an $m \times m$ square.

It is worth emphasizing that we evaluate DrMoM in a generic setting without making assumptions that are favorable to DrMoM. This serves as a basis for a fair comparative performance study between DrMoM and an existing location-based routing protocol presented in Section 9.3. It is clear from the above description that DrMoM has significant advantages over existing protocols when assuming a geographical distribution and traffic pattern for which it is often practically the case that the sources and destinations are not far away.

### 9.2.2 Mathematical Model

According to [113], as a mobile node moves, the rate $\sigma$ at which it crosses local region boundaries can be calculated as:

$$\sigma = \frac{v\pi}{4R_l} \tag{9.1}$$

Because a home region location update is triggered every time a local region boundary crossing occurs, the rate of home region location updates $\lambda_h$ is equal to $\sigma$. Local region location updates are triggered whenever the distance between the current location and the location where the last update happened exceeds the threshold $\tau$, which is equal to the wireless transmission range. Thus, the rate of local region location updates $\lambda_l$ of a mobile node depends on the wireless transmission range $r$ and the moving speed $v$ of the node, computed as follows:

$$\lambda_l = \frac{v}{r} \tag{9.2}$$
The broadcast cost $b(R)$ in a region with radius $R$ is defined as the number of wireless transmissions to cover the entire region, and can be approximated as follows [113]:

$$b(R) = 1 + \frac{\pi R^2}{\pi r^2} = 1 + \frac{R^2}{r^2} \quad (9.3)$$

The average distance $\bar{d}$ between any mobile node and its home region in the $m \times m$ square area can be estimated as [48]:

$$\bar{d} = \frac{2m}{3} \quad (9.4)$$

Therefore, the average number of hops $\alpha$ between any mobile node and its home region in the $m \times m$ square area can be approximated as follows:

$$\alpha = \frac{\bar{d}}{r} \quad (9.5)$$

**Location Update Cost $C_u$**

The location update cost $C_u$ consists of two parts: $C^l_u$, the cost for local region location updates, and $C^h_u$, the cost for home region location updates. A local region location update from a mobile node $S$ requires broadcasting the location update message among the neighbors in $S$’s local region, thus incurring a broadcast cost of $b(R_l)$. A home region location update requires sending the location update message to $S$’s home region that incurs a cost of $\alpha$, followed by a broadcast of the message within $S$’s home region that adds a broadcast cost of $b(R_h)$. Therefore, $C^l_u$ and $C^h_u$ are calculated respectively as follows:

$$C^l_u = b(R_l)$$
$$C^h_u = \alpha + b(R_h) \quad (9.6)$$

**Location Query Cost $C_q$**

The location query cost $C_q$ consists of the cost for a local region location query and optionally the cost for a home region location query which happens only when the local region location query fails. Let $C^l_q$ and $C^h_q$ denote the cost for a location region location query and the cost for a home region location query, respectively. Let $p^h_q$ denote the probability that the home region location query is needed to locate the target mobile node $D$, i.e., $p^h_q$ is the probability that the local region location query fails. $C_q$ is calculated as follows:

$$C_q = C^l_q + p^h_q \cdot C^h_q \quad (9.7)$$

A local region location query requires broadcasting the location query message among the neighbors in the local region of the source mobile node $S$, and collecting replies from these neighbors.
Therefore, the cost for a local region location query consists of the broadcast cost \( b(R_l) \) in the source mobile’s local region and the cost for the neighbors who keep valid location information of \( D \) to send the relies back to \( S \). The number of neighbors in \( S \)'s local region who keep the location information of \( D \) can be estimated based on the node density. Specifically, a neighbor in \( S \)'s local region keeps the updated location information of \( D \) when it is also within \( D \)'s local region or home region, the probability of which is \( \frac{\pi R_l^2 + \pi R_h^2}{m^2} \), assuming that the \( n \) mobile nodes are uniformly distributed in the network. Therefore, the number of neighbors who keep the location information of \( D \) can be estimated as follows:

\[
\frac{\pi R_l^2 + \pi R_h^2}{m^2} \cdot \pi R_l^2 \cdot \gamma
\]

Given the estimated number of neighbors in \( S \)'s local region who have the location information of \( D \), \( C_l^v \) can thus be estimated as:

\[
C_l^v = b(R_L) + \frac{\pi R_l^2 + \pi R_h^2}{m^2} \cdot \pi R_l^2 \cdot \gamma
\]

A home region location query requires sending the location query message to \( D \)'s home region, followed by forwarding the location reply back to \( S \). Therefore, the cost for the home region location query \( C_h^q \) consists of the costs for sending the location query message and location reply message, calculated as follows:

\[
C_h^q = 2\alpha
\]

\( S \) needs to initiate a home region location query only if the local region location query fails when none of the mobile nodes in \( S \)'s local region could find a valid entry for \( D \) in their \( LT_l \) and \( LT_h \). A mobile node in \( S \)'s local region could not find a valid entry for \( D \) if it’s not in \( D \)'s local region and home region, the probability of which is \( 1 - \frac{\pi R_l^2}{m^2} - \frac{\pi R_h^2}{m^2} \). \( p_h^q \) is the probability that all nodes in \( S \)'s local region are not in \( D \)'s local region or home region, which is computed as follows:

\[
p_h^q = (1 - \frac{\pi R_l^2}{m^2} - \frac{\pi R_h^2}{m^2}) \pi R_l^2 \cdot \gamma
\]

**Data Packet Delivery Cost \( C_d \)**

As discussed in Section 9.1.4, depending on whether there is a valid entry for the target mobile node in the source mobile node’s location tables \( LT_l \) or \( LT_h \), there could be three cases to consider. In the first two cases where a valid entry is found in \( LT_l \) or \( LT_h \), data packet delivery follows Algorithm 6. In the third case, however, a location query needs to be performed to first locate the target mobile node before data packets can be delivered using Algorithm 6. Let \( C_d^1 \) and \( C_d^2 \) denote
the cost for data packet delivery for the first two cases using Algorithm 6. Also let \( p_1 \) and \( p_2 \) denote the probability that a valid entry is found in \( LT_l \) and the probability that a valid entry is found in \( LT_h \), respectively, then \( C_d \) is calculated as:

\[
C_d = p_1 \cdot C_d^1 + p_2 \cdot C_d^2 + (1 - p_1 - p_2) \cdot C_q
\] (9.12)

Data delivery in the first case only involves mobile nodes in \( S \)'s local region that make progress moving data packets towards \( D \), and the distance from \( S \) to \( D \) is bound by the diameter of the region \( 2R_l \). Therefore, we can estimate an upper bound of \( C_d^1 \) as follows:

\[
C_d^1 = \frac{2R_l}{r}
\] (9.13)

Data delivery in the second case consists of two stages: the first stage routes the data packet from \( S \) to the first mobile node (say \( X \)) on the route that is within \( D \)'s local region, and the second stage is equivalent to data delivery in the first case, except that the source mobile node is \( X \). Therefore, we can estimate \( C_d^2 \) as follows:

\[
C_d^2 = \alpha + C_d^1
\] (9.14)

The source mobile node \( S \) can find a valid entry in either \( LT_l \) or \( LT_h \) only if \( S \) is within the local region or home region of \( D \). The probability \( p_1 \) (\( p_2 \)) that \( S \) is within the local region (home region) of \( D \) can be calculated as follows, assuming that the \( n \) mobile nodes are evenly distributed in the MANET:

\[
\begin{align*}
p_1 &= \frac{\pi R^2}{R^2} \\
p_2 &= \frac{\pi R^2}{m^2}
\end{align*}
\] (9.15)

### 9.2.3 Home Region Maintenance Cost \( C_m \)

As discussed above, DrMoM handles the case that a mobile node \( B \) enters into the home region of another node \( A \) and becomes a location server for \( A \) by requiring each node in \( A \)'s home region to periodically broadcast an announcement message to its neighbors within its wireless transmission range. This incurs a home region maintenance cost \( C_m \), consisting of the cost incurred for one wireless transmission by each node in the home region. Therefore, the calculation of \( C_m \) is shown as follows:

\[
C_m = \pi R^2 \cdot \gamma
\] (9.16)
9.2.4 Total Communication Cost $C$

The total communication cost consists of the data packet delivery cost ($C_d$), the location update cost ($C_u$), the location query cost ($C_q$, which is contained in the data delivery cost), and the home region maintenance cost ($C_m$), multiplied by their rates respectively. $C$ is calculated as follows:

$$C = \phi \cdot C_d + \lambda_l \cdot C_u^l + \lambda_h \cdot C_u^h + \mu \cdot C_m \quad (9.17)$$

9.3 Performance Evaluation

We consider a scenario that $n$ mobile nodes are evenly distributed in an area of dimensions 2000$m$ by 2000$m$. $n$ varies from 100 to 800 with an increment of 100, so that the density of nodes is a function of $n$. The wireless transmission range is 200$m$. We model the data stream between a source and a destination using a constant-bit-rate (CBR) stream at a rate of 50 packets/s. The moving speed of mobile nodes varies between $1m/s$ to $20m/s$.

![Figure 9.3: Total communication cost vs. $R_l$ in DrMoM.](image1)

![Figure 9.4: Total communication cost vs. $R_h$ in DrMoM.](image2)
We first evaluate the effect of $R_l$ ($R_h$) on the performance of DrMoM by varying the value of $R_l$ ($R_h$) but keeping $R_h$ ($R_l$) fixed. Fig. 9.3 and Fig. 9.4 show the total communication cost as a function of $R_l$ and $R_h$, respectively, for a scenario where $n = 100$ and $v = 2m/s$. As can be seen in the figures, both $R_l$ and $R_h$ are key parameters and have a significant effect of the total communication cost incurred by DrMoM. More importantly, there exists optimal $R_l$ ($R_h$) that minimizes the total communication cost incurred by DrMoM. Increasing $R_l$ of a mobile node (and thus the area of the local region) increases the chance that the node is located utilizing only local location information, but it also increases the location update cost as well as the data delivery cost because a data packet tends to travel a longer distance in the local region after it reaches the first node within the local region. The same reasoning applies to $R_h$.

![Total communication cost vs. $R_l$ and $R_h$ in DrMoM.](image)

Figure 9.5: Total communication cost vs. $R_l$ and $R_h$ in DrMoM.

Fig. 9.5 further shows the total communication cost incurred by DrMoM as a function of both $R_l$ and $R_h$. The figure depicts the effect of the interaction between $R_l$ and $R_h$ on the total communication cost incurred by DrMoM, and it justifies that there exists an optimal combination of $R_l$ and $R_h$ that minimizes the total communication cost incurred by DrMoM. It can also be seen in the figure that the total communication cost increases sharply when $R_l$ and/or $R_h$ are too large or too small.

### 9.3.1 Performance Comparison

In this section, we compare DrMoM with a location-based routing protocol called SLURP [113]. SLURP handles mobility management using a scalable location service based on statically partitioned and assigned home regions. When a mobile node moves, it updates its location with the location servers in its home region by sending location update messages. To locate a destination mobile node $D$, the node’s home region is queried to locate the home region in which $D$ currently
resides. Geographical routing is used to forward a data packet sent to $D$ towards the center of the local region of $D$. When the data packet arrives at the first node within the local region, Dynamic Source Routing (DSR) is employed to deliver the data packet to $D$ within the region.

SLURP defines the region size statically when the coverage area of a MANET is partitioned into grids, each of which corresponds to a region. This can be interpreted as having statically and equally sized home regions and local regions in DrMoM. Therefore, SLURP can be viewed as a special case of DrMoM. Compared with SLURP, DrMoM not only dynamically determines the optimal home region size and local region size defined by the radii $R_h$ and $R_l$ to minimize the total communication cost incurred, it also utilizes local regions for local location queries to increase the chance that a destination mobile node can be located using only local location information from neighbors and consequently to reduce the frequency of costly home region location queries.

To make a fair comparison between DrMoM and SLURP, we use the same parameter values as reported in [113] and evaluate their performance under identical settings. Fig. 9.6 compares the total communication cost incurred per time unit by SLURP and DrMoM as a function of the moving speed $v$. As can be seen in the figure, DrMoM under the optimal setting (optimal $R_l$ and $R_h$ that together minimize the total communication cost) outperforms SLURP over a wide range of moving speed. This result clearly demonstrates the benefit of dynamically determining the optimal $R_l$ and $R_h$ for network cost minimization in DrMoM.

![Figure 9.6: Total communication cost vs. $v$ between SLURP and DrMoM.](image)

Fig. 9.7 compares the total communication cost incurred per time unit by SLURP and DrMoM as a function of the total number of mobile nodes $n$, or equivalently the node density. As the figure illustrates, DrMoM is superior to SLURP in terms of the total communication cost incurred per time unit. The advantage of DrMoM over SLURP is particularly significant when the node density
is relatively small. Again, the figure shows that the node density is a key parameter that affects the total communication cost incurred by both DrMoM and SLURP.

Figure 9.7: Total communication cost vs. $n$ between SLURP and DrMoM.

9.4 Summary

In this chapter, we proposed and analyzed a scalable mobility management scheme for location-based routing in MANETs called DrMoM based on the design notion of integrated mobility and service management for network cost minimization. Specifically, unlike existing location services that define the home region size statically at design time, DrMoM dynamically determines the optimal home region size and local region size (defined by their respective radii denoted by $R_h$ and $R_l$), which together minimize the overall network cost incurred by mobility management and data packet delivery. We showed that the two key design parameters $R_h$ and $R_l$ as well as the total communication cost incurred by DrMoM can be derived using the mathematical model developed in the chapter, given system parameters characterizing the mobility and service characteristics of mobile nodes. We carried out a comparative performance study between DrMoM and SLURP that handles mobility management using statically assigned home regions, and our results showed that DrMoM operating under the optimal setting outperforms SLURP.
Chapter 10

Applicability and Implementation

In this chapter we discuss issues and solutions related to the applicability and implementation of the protocols proposed in this dissertation, taking into consideration that real mobile devices can be highly diverse with respect to their computing power and storage capacity.

10.1 Computation of Optimal Protocol Settings

For each protocol, an MC needs to determine the optimal protocol setting to apply at runtime in order to minimize the network cost incurred, given as input a set of input parameters characterizing an MC’s mobility and service behavior and the network condition. The optimal setting of a protocol refers to the optimal values of design parameters, e.g., \( K \) of the static and dynamic anchor schemes in Chapter 4 and \( R_l \) and \( R_h \) of DrMoM in Chapter 9.

We investigate two approaches for determining optimal protocol settings, namely, computation-based and table lookup based. The computation-based approach executes a computational procedure developed for each protocol based on the mathematical model of the protocol (e.g., the SPN model in Figure 8.4 of Chapter 8 for HASRM). The core of the computational procedure is to apply a cost function or a set of cost functions specific to the protocol to compute the total communication cost given a set of input parameter values measured at runtime. For example, Equations 4.8 and 4.9 in Chapter 4 are such functions that calculate \( C_{\text{static}} \) and \( C_{\text{dynamic}} \), the total communication costs incurred by the static and dynamic anchor schemes, respectively. The computation-based approach essentially determines the optimal protocol setting by searching for the best design parameter values that minimize the total cost.

The table lookup based approach uses a lookup table that stores the optimal protocol setting over perceivable ranges of input parameter values. At runtime, the optimal protocol setting can be determined by performing a simple table lookup operation using input parameter values measured
at runtime as indices to the table. Compared with the computation-based approach, this approach needs little computational time at runtime to determine the optimal protocol setting, whereas it needs a considerable amount of memory to store the lookup table. Choosing which approach to use is a tradeoff between time and space. In the next section, we discuss the feasibility of these two approaches in terms of the computational time and memory space required.

It is worth noting that the computation of optimal protocol settings is highly scalable, as each MC computes its optimal protocol setting on a per-user basis without coordinating with other MCs or relying on global information.

10.2 Feasibility of Determining Optimal Protocol Settings at Runtime

We evaluate the feasibility of each of the above two approaches by the amount of computational time and/or the amount of memory space it uses at runtime. It depends on a number of factors, including:

- The method for finding the optimal values of design parameters that minimize the total communication cost.
- The number of design parameters (e.g., the local region radius $R_l$ and home region radius $R_h$ in Chapter 9).
- The number of input parameters (e.g., the mobility rate $\sigma$ and data query rate $\lambda$ in Chapter 6).
- The width of perceivable parameter value ranges.
- The granularity of discretization of parameter spaces.
- The tolerance to inaccuracy of the result.

For the computation-based approach that executes a computational procedure at runtime, these factors mainly affect the amount of computation time needed to determine the optimal protocol setting. For the table lookup approach, these factors mainly affect the amount of memory needed for storing the lookup table. The feasibility of the computation-based approach depends on the amount of computational time needed to find the optimal protocol setting. The amount of computational time needed depends on the search method used. We will further discuss the tradeoff between computational time vs. solution quality in Section 10.3.
The feasibility of the table lookup based approach that performs a table lookup to determine the optimal protocol setting depends on how much memory space the table occupies. If the number of input parameters is large and each input parameter spans a considerably wide range, the lookup table will need a considerable amount of memory space, which may be excessive for mobile devices. To trade accuracy off for memory space saving, a large discretization factor may be used to reduce the space dimension, thereby reducing the amount of memory used for storing the lookup table. Extrapolation or interpolation techniques then can be used when input parameter values do not map directly to the stored values in the table. The computational cost of the table lookup based approach is minimum as it only involves a table lookup operation augmented with extrapolation or interpolation to find the optimal setting.

10.3 Heuristic Search

In this section we investigate heuristic search methods and their effect on reducing the amount of computational time for identifying the optimal protocol setting at runtime using the computation-based approach. Reducing the amount of computational time needed by the computation-based approach is especially important when the table lookup approach is not feasible due to excessive memory usage for storing the lookup table.

We setup the problem of determining the optimal protocol setting that minimizes the total communication cost as an optimization problem as follows:

$$y^* = \min_{x \in S} f(x)$$

where $x$ is a vector of $n$ design parameter values (one for each design parameter), $S$ is an $n$-dimensional feasible region covering the perceivable ranges of design parameter values, and $f$ is the cost objective function, defined over $S$. The objective is to find $x^*$ in $S$ that minimizes $f$, i.e., $y^* = f(x^*)$. For protocols proposed in this dissertation, the number of design parameters, i.e., $n$, is less than or equal to 2, resulting in a limited amount of computational overhead at runtime.

We study two representative heuristic search methods here, namely, downhill simplex [119] (also called the Nelder-Mead method) and simulated annealing [120]. The Nelder-Mead method is a commonly used nonlinear heuristic search method for minimizing a given objective function in a multi-dimensional search space. The method approximates a local minimum of the objective function of $N$ decision variables, using a $(N + 1)$-vertex simplex. Simulated annealing is a well-known heuristic search method for approximating the global optimum of a given objective function.
over a large search space. Simulated annealing is known for its capability to find a near-optimal solution close to the global optimum of a hard problem (e.g., NP-complete problems) in a reasonable amount of time. Both methods are lightweight in terms of memory usage as they evaluate the given objective function iteratively until the convergence condition is satisfied or the number of iterations reaches the threshold.

We exemplify the problem of determining the optimal protocol setting with the DrMoM protocol proposed in Chapter 9. The optimal protocol setting for DrMoM consists of two design parameters, namely, $R_l$ and $R_h$, representing the local region radius and home region radius, respectively. Therefore, the search space is a two-dimensional plane. The cost objective function $f$ is given by Equation 9.17 in Chapter 9 for calculating the total communication cost incurred by DrMoM. We use brute-force search as a baseline here to demonstrate the benefit of heuristic search methods. For brute-force search, a discrete two-dimensional grid (with regularly spaced points) is defined over the search space, and the objective function is evaluated at every point of the grid to determine the minimum. We evaluate the brute-force method using two grids of different scales. Specifically, one grid has a spacing of 1 meter ($d = 1$), while the other has a spacing of 10 meters ($d = 10$). The larger the spacing between points, the smaller the number of points to evaluate and therefore the less amount of computational time needed, at the expense of accuracy.

Table 10.1: Computational time (in seconds) needed to find a solution by various search methods (BF stands for Brute-Force).

<table>
<thead>
<tr>
<th></th>
<th>BF ($d = 1$)</th>
<th>BF ($d = 10$)</th>
<th>Nelder-Mead</th>
<th>Simulated Annealing</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n = 100, v = 2.0$</td>
<td>4.548</td>
<td>0.0480</td>
<td>0.0536</td>
<td>0.191</td>
</tr>
<tr>
<td>$n = 100, v = 8.0$</td>
<td>4.569</td>
<td>0.0486</td>
<td>0.0532</td>
<td>0.229</td>
</tr>
<tr>
<td>$n = 200, v = 2.0$</td>
<td>4.637</td>
<td>0.0480</td>
<td>0.0426</td>
<td>0.174</td>
</tr>
<tr>
<td>$n = 200, v = 8.0$</td>
<td>4.535</td>
<td>0.0484</td>
<td>0.0470</td>
<td>0.210</td>
</tr>
<tr>
<td>$n = 400, v = 2.0$</td>
<td>4.491</td>
<td>0.0484</td>
<td>0.0426</td>
<td>0.199</td>
</tr>
<tr>
<td>$n = 400, v = 8.0$</td>
<td>4.495</td>
<td>0.0492</td>
<td>0.0530</td>
<td>0.198</td>
</tr>
<tr>
<td>$n = 800, v = 2.0$</td>
<td>4.520</td>
<td>0.0486</td>
<td>0.0574</td>
<td>0.224</td>
</tr>
<tr>
<td>$n = 800, v = 8.0$</td>
<td>4.381</td>
<td>0.0482</td>
<td>0.0478</td>
<td>0.200</td>
</tr>
</tbody>
</table>

To implement the above methods, we use the Python library SciPy [121], which is a widely used open-source library for scientific computing in Python. SciPy includes a package named scipy.optimize for a wide variety of optimization and root finding problems. We use three functions in the package, namely, fmin, anneal, and brute, to implement the Nelder-Mead method, the simulated annealing method, and the brute-force method, respectively.
Table 10.1 compares the computational time (average over 10 samples) needed to find a solution, by the various search methods, given the node population \( n \) and moving speed \( v \) as input. Table 10.2 shows the values of \( R_l \) and \( R_h \) found by the various search methods. Correspondingly, Table 10.3 shows the minimum cost obtained by these methods.

It can be seen that the Nelder-Mead method shows the best tradeoff among the methods investigated. Specifically, the Nelder-Mead method produces the best minimum total communication cost while requiring the minimum computational time. Simulated annealing also produces the best minimum total communication cost but uses more computational time on average than the Nelder-Mead method. The brute-force method, either fine-grained \((d = 1)\) or coarse-grained \((d = 10)\), performs worst than Nelder-Mead or simulated annealing in terms of the minimum total communication cost. The fine-grained brute-force method also requires the largest amount of computational time among all to find a solution. The result demonstrates that heuristic search achieves a well-balanced tradeoff between the computational time needed to find a solution and the optimality of the solution. The fact that the Nelder-Mead method and simulated annealing produce the same minimum implies that the local minimum (approximated by the Nelder-Mead method) is possibly the global minimum (approximated by simulated annealing).

### 10.4 Measurement of Input Parameter Values

In the computation-based approach, the objective function to be minimized takes a number of input parameters. Examples of input parameters include the mobility rate, data packet rate, data object query and update rates, etc. The protocols proposed in this dissertation leverage cross-layer design to obtain input parameter values characterizing a mesh client’s mobility and service characteristics.
Table 10.3: The minimum communication cost found by various search methods (BF stands for Brute-Force).

<table>
<thead>
<tr>
<th></th>
<th>BF ($d = 1$)</th>
<th>BF ($d = 10$)</th>
<th>Nelder-Mead</th>
<th>Simulated Annealing</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n = 100, v = 2.0$</td>
<td>211.403</td>
<td>211.952</td>
<td>211.139</td>
<td>211.139</td>
</tr>
<tr>
<td>$n = 100, v = 8.0$</td>
<td>211.061</td>
<td>212.175</td>
<td>211.355</td>
<td>211.355</td>
</tr>
<tr>
<td>$n = 200, v = 2.0$</td>
<td>174.877</td>
<td>175.558</td>
<td>174.707</td>
<td>174.707</td>
</tr>
<tr>
<td>$n = 200, v = 8.0$</td>
<td>175.082</td>
<td>175.762</td>
<td>174.914</td>
<td>174.914</td>
</tr>
<tr>
<td>$n = 400, v = 2.0$</td>
<td>145.229</td>
<td>145.373</td>
<td>145.100</td>
<td>145.100</td>
</tr>
<tr>
<td>$n = 400, v = 8.0$</td>
<td>145.430</td>
<td>145.569</td>
<td>145.304</td>
<td>145.304</td>
</tr>
<tr>
<td>$n = 800, v = 2.0$</td>
<td>122.112</td>
<td>122.711</td>
<td>122.009</td>
<td>122.009</td>
</tr>
<tr>
<td>$n = 800, v = 8.0$</td>
<td>122.315</td>
<td>122.506</td>
<td>122.212</td>
<td>122.212</td>
</tr>
</tbody>
</table>

and network conditions.

Specifically, the mobility rate of an MC can be estimated periodically by counting the number of serving MR changes during a fixed interval, say, every 30 minutes. Implementation wise, an MC maintains a counter for the number of serving MR changes, and the counter is incremented whenever the MC changes its serving MR, which can be detected by the MC at the link layer. At the end of each interval, the mobility rate is calculated and the counter is reset. An MC can dynamically estimate its data packet rate by periodically monitoring the sequence numbers of data packets at the network layer. Finally, the data object update/query rate can be dynamically estimated by counting the number of data object queries/uploads periodically at the application layer (the definition of data objects updates/queries is application specific).

### 10.5 Application of the Optimal Protocol Setting

After the optimal protocol setting is determined, the next step is to apply the optimal protocol setting at runtime to make it effective. For some protocols, applying the optimal protocol setting can be done solely by the MC. Specifically, under the dynamic anchor protocol for integrated mobility and service management (Chapter 4), an MC keeps track of the current forwarding chain length, and compares it with the optimal threshold $K$ for the forwarding chain length determined at runtime. If the current optimal threshold is reached, it simply resets the forwarding chain and registers its current serving MR as its new AMR by sending a location update message to update the location database at the gateway.

For some protocols, the application of the optimal protocol setting requires cooperation between an MC and MRs. Specifically, under the DAHM protocol for mobile multicast services (Chapter 7),
an MC keeps track of its distance (number of hops) to its current MA and compares it with the optimal threshold $H$ for the number of hops a member can be away from its MA. If the current optimal threshold is reached, the MC selects its current serving MR as its new MA and the new MA joins the multicast backbone as a new tree node if necessary. In this case, the MC and its new MA work together to apply the optimal protocol setting.

### 10.6 Implementation on Top of Existing Network Protocols

In this section we discuss how the proposed protocols can be built on top of existing network protocols such as IP. As discussed above, all the proposed protocols in this dissertation measure the values of input parameters such as the mobility rate and data packet rate. This requires interaction among different network layers. Therefore, cross-layer design needs to be applied to the proposed protocols.

The mobility management protocols, i.e., the static anchor and dynamic anchor protocols proposed in Chapter 4 and the LMMesh protocol proposed in Chapter 5, operate at the network layer on the MR side. This is because these protocols setup forwarding pointers between routers on the forwarding chain, which can be implemented by routing table entries with appropriate values in the “next hop” field. To implement these mobility management protocols, the network layer protocol (IP for example) needs to be slightly modified to make it aware of and capable of processing control packets for setting up forwarding pointers. Specifically, the protocols introduce a new type of control packets in addition to existing ones such as ICMP to carry information about the forwarding pointer to be setup between two MRs.

The APPCCM protocol proposed in Chapter 6 for integrated mobility and cache consistency management for mobile data access in WMNs runs at the application layer on the MR side. This is because MRs involved in the operation of APPCCM serve as data proxies that cache application layer data objects accessed by MCs. There is an application layer server daemon running on each MR serving as a data proxy, which maintains the cached data objects, services requests for cached data objects from the MCs, participates in the migration of cached data objects to other data proxies, and requests data objects from the source on behalf of the MCs when cached copies cannot be found. To make it possible for MCs to dynamically select MRs as data proxies, the server daemon runs on every MR (regardless of whether they currently serve as data proxies or not). It should be noted that MRs, as routers, inherently perform routing functions at the network layer. APPCCM requires MRs to also run the server demon at the application layer.
Protocols proposed for mobile multicast in WMNs, i.e., the DAHM protocol proposed in Chapter 7 and the HASRM protocol proposed in Chapter 8, run on the network layer or the application layer, depending on which layer the multicast structure is maintained. If the multicast structure is maintained as an application layer overlay, the protocols run on the application layer on the MR side; otherwise they run on the network layer. Note that although we assume that an SPT is used to construct the multicast backbone, both DAHM and HASRM also work with other types of multicast tree structures. For the case in which the multicast structure is maintained as an application layer overlay, it does not need any modification to the underlying existing network protocols. Specifically, there is an application layer server daemon running on every MR on the multicast tree responsible for relaying multicast data. The server daemon running on an MR serving as an MA is also responsible for managing local multicast group membership information and forwarding multicast data to multicast members in the local group. Here again we note that MRs, as routers, inherently perform routing functions at the network layer. DAHM and HASRM require MRs to also run the server daemon at the application layer.
Chapter 11

Conclusion

11.1 Publications

This dissertation work has resulted in three journal papers (based on Chapters 4, 5, and 6), two conference papers (based on Chapters 6 and 8), and two journal submissions (based on Chapters 7, 9).

11.1.1 Papers Published


11.1.2 Papers Submitted


11.2 Summary and Future Work

In this dissertation research, we have proposed and investigated a new design notion of integrated mobility and service management for WMNs from which several protocols were developed and analyzed both analytically and by simulation. A comprehensive analysis comparing these mobility and service management protocols with existing protocols to-date has been conducted. The results demonstrated the superiority of our integrated mobility and service management protocol design concept. Two mobility management schemes for WMNs based on pointer forwarding, namely, the static anchor and dynamic anchor schemes were discussed in Chapter 4. In Chapter 5 we introduced LMMesh for integrating routing-based location update with pointer forwarding for mobility management. In Chapter 6, we developed APPCCM for cache consistency management in mobile data access in WMNs. In Chapter 7, we developed DAHM for supporting efficient mobile multicast in WMNs. The HASRM algorithm proposed for secure and reliable mobile multicast in WMNs was introduced in Chapter 8. Lastly in Chapter 9 we extended the design notion of integrated mobility and service management to MANETs by developing DrMoM and demonstrating its superiority over existing home-region based location management protocols.

There are several future research directions as extensions to this dissertation research:

1. **Mobility Management in WMNs with Multiple Gateways:** The presence of multiple gateways affects the protocol design in all aspects, including reliability, availability, and throughput considerations. In addition, the protocol design also needs to consider load balancing among the multiple gateways. For example, dynamic anchor based mobility management proposed in Chapter 4 and LMMesh proposed in Chapter 5 can take the advantage of multiple gateways to balance the load of mobility management tasks among the gateways so that the gateways are not the performance bottleneck, nor a single point of failure.
2. **Mobility Management in hybrid WMNs**: In a hybrid WMN [1], MCs may form ad hoc networks interconnected with the mesh backbone. Consequently some MCs have to rely on other MCs to access the mesh backbone because they are not close to any MR. Because MCs are mobile, pointer forwarding is no longer an appropriate solution because the forwarding chain can be easily broken as MCs move. Therefore, a new mobility management protocol that works appropriately regardless how an MC is connected (either directly or through other MCs) to the mesh backbone is needed for hybrid WMNs.

3. **Mobile Data Access in WMNs**: APPCCM can be enhanced further to support community-based sharing of data objects. Specifically, APPCCM can provide more efficient sharing of data objects when (a) multiple MCs share the same data proxy, and (b) multiple MCs access the same data object cached at distinct data proxies. With this capability, an MC can potentially access cached data objects stored by the data proxy for which it does not keep a cached copy. Also several copies of the same object may be consolidated into one copy for efficient data sharing. Another direction is to devise efficient integrated cache invalidation and replacement management that can further reduce the data access latency and the overall network cost. We also plan to investigate cache replacement policies that take into consideration varying data object sizes, query/update frequency, and locality.

4. **Multicast in WMNs**: There are several directions for multicast research. One direction is to extend our multicast protocol design to support multiple multicast groups in a WMN. Another direction is to use an MC as an MA when a group member cannot find a nearby MR and must rely on other MCs for network traffic relaying. Having MCs serve as MAs extends the coverage area of a multicast group to those members that are not connected to any MR. For secure and reliable mobile multicast, a direction is to extend HASRM to handle failures of MRs and MAs in addition to packet losses on wireless links to provide a complete solution to secure and reliable multicast in WMNs. Lastly, it is worth investigating methods for efficiently supporting source mobility.

5. **Integrated mobility and service management in MANETs**: A future direction is applying the same protocol design principle for WMNs to MANETs in supporting mobile data access and multicast services. Another direction is to investigate the possibility of using spatial index to speed up location services in DrMoM. Spatial index such as R-tree or kd-tree organizes spatial objects hierarchically to support efficient queries of spatial objects and
distances between spatial objects. With the help from spatial index, DrMoM can efficiently and accurately answer a query for the location of a mobile node for location services in MANETs.
Bibliography


Summary of Acronyms

**WMNs** Wireless Mesh Networks

**MANETs** Mobile Ad hoc Networks

**WSNs** Wireless Sensor Networks

**APPCCM** Adaptive Per-user Per-object Cache Consistency Management

**DAHM** Dynamic Agent-based Hierarchical Multicast

**MA** Multicast Agents

**HASRM** Hierarchical Agent-based Secure and Reliable Multicast

**EIA/TIA** Electronic and Telephone Industry Associations

**IS-41** Interim Standard 41

**GSM** Global System for Mobile Communications

**MAP** Mobile Application Part

**MSC** Mobile Switch Center

**RA** Registration Area

**MTs** Mobile Terminals

**VLR** Visitor Location Register

**HLR** Home Location Register

**MIP** Mobile IP

**MN** Mobile Node
HA  Home Agent
FA  Foreign Agent
CN  Correspondence Node
CoA  Care-of Address
MIPv6  Mobile IP v6
MIP-RR  Mobile IP Regional Registration
GFAs  Gateway Foreign Agents
HMIP  Hierarchical Mobile IP
IDMP  Intra-Domain Mobility management Protocol
GCoA  Global Care-of Address
LCoA  Local Care-of Address
CIP  Cellular IP
HAWAII  Handoff Aware Wireless Access Internet Infrastructure
DSDV  Destination-Sequenced Distance Vector
OLSR  Optimized Link State Routing
FSR  Fisheye State Routing
WRP  Wireless Routing Protocol
AODV  Ad hoc On-demand Distance Vector
DSR  Dynamic Source Routing
ABR  Associativity-Based Routing
LMR  Lightweight Mobile Routing
DREAM  Distance Routing Effect Algorithm for Mobility
GLS  Grid Location Service
IR  Invalidation Report
CB  CallBack
UAR  Update-to-Access Ratio
PER  Poll Each Read
RS  Remote Subscription
BT  Bi-directional Tunneling
RBMoM  Range-Based Mobile Multicast
MHA  Multicast Home Agent
MMAs  Mobile Multicast Agent
SPT  Shortest-Path Tree
MCT  Minimum-Cost Tree
HRM  Hierarchical Reliable Multicast
AMR  Anchor Mesh Router
SPN  Stochastic Petri Net
SMR  Service to Mobility Ratio
CCM  Client-Cache Mode
DPM  Data-Proxy Mode
QUR  Query to Update Ratio
RRM  Regional-Registration based Multicast
DTM  Dynamic Tree-based Multicast
SPT  Shortest-Path Tree
SLURP  Scalable Location Update based Routing Protocol
MFR  Most Forwarding with fixed Radius
SNR  Signal-to-Noise Ratio