

國立交通大學

資訊工程學系

博士論文

階層式行動網路下之位址分配、封包繞送、以及
資訊存取機制

Addressing, Routing, and Information Provisioning
Mechanisms for Hierarchical Mobile Networks

研究生：徐元瑛

指導教授：曾建超 教授

中華民國九十四年六月

階層式行動網路下之位址分配、封包繞送、以及資訊存取機制
Addressing, Routing, and Information Provisioning Mechanisms for
Hierarchical Mobile Networks

研究生：徐元瑛

Student : Yuan-Ying Hsu

指導教授：曾建超

Advisor : Chien-Chao Tseng

國立交通大學
資訊工程學系
博士論文



A Dissertation Submitted to
Department of Computer Science and Information Engineering
College of Electrical Engineering and Computer Science
National Chiao Tung University
In Partial Fulfillment of the Requirements
for the Degree of
Doctor of Philosophy
in
Computer Science and Information Engineering

June 2004

Hsinchu, Taiwan, Republic of China

中華民國九十四年六月

階層式行動網路下之位址分配、封包繞送、以及資訊存取機制

研究生: 徐元瑛

指導教授: 曾建超 博士

國立交通大學資訊工程學系博士班

摘要

由於無線網路以及無線裝置的技術更新，依不同的需求已有各式各樣的無線網路因應而生，例如行動隨意網路（mobile ad-hoc networks）、無線感測網路（wireless sensor networks）、無線網狀網路（wireless mesh networks）、以及移動網路（mobile networks）。一無線網路可以整體移動，也可與其他無線網路重疊進而形成階層式移動網路。階層式移動網路提供移動節點一個彈性的方式使其可以利用適合的無線存取技術連接到基礎網路。但是，當移動節點在階層式移動網路中漫遊時，仍有許多技術問題需要解決。在這篇論文中，我們提出了一些在階層式移動網路中位址分配、封包繞送、以及資訊存取的方法。

現有的位址分配技術通常採用廣播的方式來尋求位址或檢查是否有重複使用，但廣播會在多點跳躍的網路中造成相當大的負擔。我們提出了一個稱為 Prime DHCP 的位址分配方式使得在分配位址的過程中不需要在整個網路中廣播訊號。Prime DHCP 將網路中每個節點設為 DHCP 代理伺服器，並且獨自執行提出的質數位址配置演算法計算出各自獨有的位址群。DHCP 代理伺服器以及質數位址配置演算法可以共同排除廣播的必要性。

根據 Prime DHCP 分配的位址結果，我們提出了一個基於質數原則的自我組態繞送協定，此協定可讓每個節點自行找到繞送路徑到其他內部節點。此方法讓每個節點只需要依據目的節點的位址即可找到繞送路徑，不需要定期與其他節點交換路由資訊或發送訊息給目的節點詢問路徑。當有封包需要送到外部網路時，我們在每一個網路中設定至少一個閘道伺服器用以負責繞送進出外部網路的封包。除此之外，我們修改了 Mobile IP 的方法以支援節點的移動。同時我們也提出了一個負擔平衡的繞送協定來平衡多個閘道伺服器的外部網路流量。

除了網路位址以及繞送的機制以外，我們也提出了在階層式網路下資訊存取的方法。當一個閘道伺服器需要服務許多移動節點時，閘道伺服器外部的網路頻寬勢必會被所有的節點一同分攤而形成瓶頸。為此我們提出了一個雙模網路架構以及一考量負載的排程機制以安排傳輸順序。此外，使用者的資訊有可能在階層式行動網路中存在各式各樣的裝置中。因此我們學習記憶體階層的方式提出了個人資訊階層（Personal Information Hierarchy, PIH）以及相對應的存取機制來存放使用者的個人資訊。

我們已針對所有提出的方法進行效能評估。評估的結果顯示 Prime DHCP 可以明顯地降低在分配位址時的時間以及所造成的訊號花費；基於質數原則的自我組態繞送協定可以減少繞送路徑的設定時間；PIH 的架構以及機制可以增加存取的空間以及使用者存取資料的速率。

關鍵字: 多點繞送無線網路、行動隨意網路、移動網路、位址分配、封包繞送、漫遊、自動組態、資訊階層。



Addressing, Routing, and Information Provisioning Mechanisms for Hierarchical Mobile Networks

Student: Yuan-Ying Hsu

Adviser: Dr. Chien-Chao Tseng

Department of Computer Science and Information Engineering

National Chiao Tung University

ABSTRACT

With the advance of wireless and terminal technologies, various wireless networks, such as mobile ad-hoc networks, wireless sensor networks, wireless mesh networks, and mobile networks, are designed for different purposes. A wireless network may move as a whole and furthermore may overlay with one another to form a hierarchical mobile network. Hierarchical mobile networks provide a flexible approach for mobile nodes to access Infrastructure networks with any appropriate wireless technologies. However, many technical issues need to be resolved for mobile nodes to roam within a hierarchical mobile network environment. In this thesis, we propose several mechanisms for network addressing, routing and information provisioning in hierarchical mobile networks.

Current address allocations usually involve broadcasting, which introduces huge overhead in multi-hop environments, for address solicitation or duplicate address detection. We propose a Prime DHCP scheme that can allocate addresses to hosts without broadcasting over the whole network. Prime DHCP makes each host a DHCP proxy and run a prime numbering address allocation algorithm individually to compute unique addresses. The concept of DHCP proxies and the prime numbering address allocation algorithm together eliminate the needs for broadcasting.

Based on the address allocation result by Prime DHCP, we propose a prime-based self-configured routing protocol for each node to route data packets to other local

nodes within the same network. With the proposed routing protocol, each node can derive a routing path to a local node according to the node's address without periodically exchanging routing information with other nodes. Furthermore, the node need not send a routing request to the destined node before forwarding packets to the local node, either. For packets destined to external networks, we configure at least one gateway in each wireless network, and have the gateways responsible for routing packets from/to external networks. To support host mobility, we adopt mobile IP with minor modifications, and we also propose a load-balanced routing protocol to balance external traffic between multiple gateways.

Besides network addressing and routing mechanisms, we also propose information provisioning mechanisms for hierarchical mobile networks. When a gateway needs to serve a lot of mobile nodes, the external bandwidth would be shared by all the mobile nodes beneath the gateway. We propose a two-tier proxy architecture and a load-based scheduling mechanism to schedule traffic according to data sizes. Furthermore, personal information of a user might be stored in various devices. Therefore we also propose a personal information hierarchy (PIH) to store personal information and corresponding information accessing policies for PIH by adapting the successful experience of memory hierarchy.

We have conducted performance evaluation for all proposed mechanisms. Performance results show that prime DHCP can significantly reduce the signal overhead and the latency for hosts to acquire addresses; prime-based self-configured routing protocol can significantly decrease path setup time and signal overhead; and the PIH architecture and accessing policies together can significantly increase the storage capacity with a negligible decrease in access speed a user can experience in personal information management.

Keywords: multi-hop wireless network, MANET, network mobility, addressing, routing, roaming, auto configuration, information hierarchy.

Acknowledgments

First of all, I would like to thank my dearest Lord Jesus for His eternal love that helps me escape from shadows and continuously have confidence in my research work.

I thank my advisor Dr. Chien-Chao Tseng for his guidance and support in the past six years. He always patiently discusses with me and gives me suggestions according to his professional experience in mobile computing. Without his assistance, I can not complete this dissertation so smoothly.

I am also grateful to Prof. Yu-Chee Tseng for his aid in system evaluation. His positive and deliberate working attitudes impress me a lot, and these characteristics are exactly what I need to learn.

I appreciate my committee, Prof. Chung-Ta King, Prof. Chyi-Ren Dow, Prof. Li-Hsing Yen, Prof. Shiao-Li Tsao, and Dr. Jen-Shun Yang, for their carefully reading my thesis and giving me valuable advice and comments.

Special thanks is due to Prof. Yi-Bing Lin for his kindly encouragement and directions of my future life.

My lovely colleagues, especially my dear classmate RT and juniors, ML, Chien-An, Yoda, and Welkin, in the Wireless Internet Laboratory are also thanked. They are the greatest team that I have ever met. Thank them to accompany me with all the joy and sorrow.

At last, I would love to thank my sweet family for their constant love and trust throughout these years. They are my most powerful back-ups.

God is love. There is no fear in love, but perfect love casts out fear.

~ 1 John 4:16, 18

Contents

Abstract in Chinese	i
Abstract in English	iii
Acknowledgments	v
Table of Contents	vi
List of Figures	ix
List of Tables	xii
1 Introduction	1
1.1 Motivation	2
1.2 Overview of proposed mechanisms	2
1.3 Synopsis	3
2 System Architecture	4
2.1 Mobile ad-hoc networks	4
2.2 Wireless sensor networks	5
2.3 Wireless mesh networks	6
2.4 Network mobility	7
2.5 Hierarchical mobile networks	8
3 Addressing Mechanisms for Hierarchical Mobile Networks	10
3.1 Previous works	11
3.1.1 Best effort allocation	11



3.1.2	Centralized allocation	11
3.1.3	Decentralized allocation	12
3.2	DHCP relay method	12
3.3	DHCP proxy method: prime DHCP	14
3.3.1	Address allocation tree	14
3.3.2	Address allocation procedures	17
3.3.3	Exception handling	21
3.3.4	Performance evaluation	22
3.4	Summary	24
4	Internal Routing Mechanisms for Hierarchical Mobile Networks	25
4.1	Motivation	25
4.2	Prime-based self-configured routing protocol	25
4.2.1	Stateless prime-based self-configured routing protocol	27
4.2.2	Stateful prime-based self-configured routing protocol	30
4.2.3	Performance evaluation	32
4.2.4	Discussion	37
4.3	Summary	41
5	External Routing Mechanisms for Hierarchical Mobile Networks	42
5.1	External Routing Architecture	42
5.2	Load-balanced routing protocol	43
5.2.1	Protocol design	44
5.2.2	Performance evaluation	45
5.3	Mobile IP Adaption	48
5.3.1	Problems of roaming in public networks	50
5.3.2	Problems of roaming in private networks	50
5.4	Location-assisted routing enhancements	52
5.4.1	Localization algorithm	52
5.4.2	Location-based fast handoff	53

5.5	Summary	54
6	Information Provisioning Mechanisms for Hierarchical Mobile Networks	56
6.1	Load-based scheduling mechanism	56
6.1.1	Two-tier architecture	57
6.1.2	Load-based scheduling scheme	58
6.1.3	Performance evaluation	63
6.2	Hierarchical personal information management	70
6.2.1	Personal information hierarchy	72
6.2.2	Information accessing policies	73
6.2.3	Service adaptation strategies	77
6.2.4	Performance evaluation	78
6.3	Summary	86
7	Conclusion and Future Works	88
	Bibliography	90

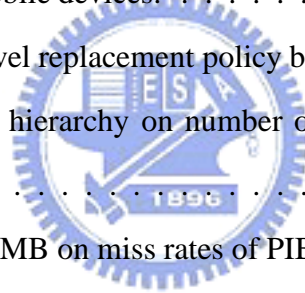


List of Figures

2.1	Two options of wireless networks: infrastructure and ad hoc.	5
2.2	An example of wireless sensor network.	5
2.3	An example of wireless mesh network.	6
2.4	An example of network which is mobile (NEMO).	7
2.5	System architecture of hierarchical mobile network.	9
3.1	Address resolution and DHCP operations for host n to join the MANET. . . .	12
3.2	The message flow of DHCP in our architecture.	13
3.3	An example of address allocation tree.	16
3.4	Message flows of the address allocation procedure of a new coming node. . .	18
3.5	An Example of MANET topology with a new coming host.	19
3.6	An example of address allocation result.	20
3.7	The address allocation tree.	20
3.8	The last two steps of Figure 3.6 when address space=20.	22
3.9	The effect of recycle period on the address utilization.	23
3.10	The effect of address space on the address utilization.	23
4.1	An example of setting routing path: (a) Address allocation tree; (b) Mesh topology; (c) Routing paths while applying stateless routing protocol; (d) Routing paths while apply stateful routing protocol.	27
4.2	The procedure of setting routing path in the source routing method of stateless routing protocol.	28
4.3	The procedure of setting routing path in the hop-by-hop routing method of stateless routing protocol.	29

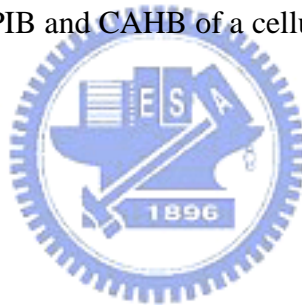
4.4	The procedure of setting routing path in the stateful routing protocol.	30
4.5	Two network topologies when network size=5*5: (a) Link=4 and (b) Link=8.	33
4.6	The signal overhead.	33
4.7	Four address allocation types: type0: from central node, type1: from left-top corner node, type2: from center-top node, and type3: from right-top corner node.	34
4.8	The average hop count.	35
4.9	The hop count difference when link=4 and type=0.	35
4.10	The hop count difference of different address assignment type when link=4. .	36
4.11	The hop count difference of different address assignment type when link=8. .	37
4.12	The address allocation tree with virtual links.	37
4.13	Example of address allocation tree with virtual links.	39
4.14	The hop count difference with limited address space.	40
4.15	The effect of levels for help on the address space requirement.	41
5.1	External routing architecture.	43
5.2	An example of load-balanced routing protocol.	45
5.3	The testing environment.	46
5.4	(a) selection of gateway by host N5, and (b) traffic loads at gateway N1 and N2.	46
5.5	Trace of N5's throughput over time.	47
5.6	Mobile IP operation scenarios.	48
5.7	An example of ping-pong routing for an MN attaches to a hierarchical mobile network.	49
5.8	Operations of NAT traversal.	51
5.9	Illustration of locating an MN.	53
5.10	An example of selecting candidate points for pre-registration.	54
6.1	The hierarchical-proxy architecture for two-tier wireless networks.	58
6.2	The load-based scheduling scheme without SPs.	59
6.3	Sequence diagram when the MONET is moving (without SPs).	60

6.4	Sequence diagram when the mobile network is moving (with SPs).	63
6.5	Completion Rates in two-tier and single high-tier networks.	65
6.6	Average waiting times in two-tier and single high-tier networks.	65
6.7	Average waiting times with different hit ratios of the GP.	66
6.8	Average waiting times with and without scheduling.	67
6.9	Completion rates with and without scheduling.	68
6.10	Performance for different ratios of moving periods to staying periods.	68
6.11	Performance for different percentages of heavyweight services.	69
6.12	Performance for different percentages of low-tier bandwidth for lightweight services.	70
6.13	Example of four-layered repositories.	71
6.14	Example of information division.	74
6.15	Memory blocks in mobile devices.	74
6.16	Algorithm of three-level replacement policy by taking e-mail for example.	77
6.17	Effect of information hierarchy on number of mails with different memory space sizes.	80
6.18	Effect of the size of IIMB on miss rates of PIB and IIMB.	81
6.19	Miss rates of PIB and IIMB with different percentage of the size of memory blocks and fixed number of mails.	81
6.20	Effect of different access behaviors on miss rates of PIB , IIMB, and CIB.	83
6.21	Miss rates of PIB and IIMB with different percentage of the size of memory blocks.	84
6.22	Effect on average access time.	85



List of Tables

3.1	Qualitative analysis of address allocation mechanisms	22
4.1	Virtual links recorded by nodes in Figure 4.12	38
5.1	The impact of traffic fluctuation to load-balancing protocol.	47
6.1	Simulation Parameters.	64
6.2	Example of abstraction for e-mail, calendar, and business card.	78
6.3	Entry sizes of IIMB, PIB and CAHB of a cellular phone.	79



Chapter 1

Introduction

Nowadays, there are various wireless networks developed for different purposes. The most common one is IEEE 802.11 Wireless LAN (WLAN), which constructs the infrastructure by access points. By deploying WLAN access points, users now may access Internet services in anytime at anywhere and with any device having wireless communication capability due to the flexibility and convenience of wireless connectivity. Mobile ad hoc network [42] further provides higher flexibility to free the necessity of base stations. Mobile hosts communicate with one another in a multi-hop manner.

Besides two basic wireless communication modes, infrastructure and ad-hoc, there are some other variations for different purposes. Network mobility [57] is addressed to provide Internet access ability for group moving like public transportation. Taking train for example, all mobile hosts on a train form a subnetwork and move together. A mobile router is deployed on the train to take responsibility of attaching to the infrastructure and serving all hosts on the train for Internet accessing services.

Another variation is wireless mesh network [26], which is recently designed to represent a promising alternative for broadband Internet access. In the past, wireless networks are limited at some particular hot spots, like coffee shops, airports, or hotels due to the difficulties of installing and maintaining a wired network backhaul connection. Wireless mesh networks replace traditional wired backhaul routers with wireless ones and all of them form a mesh topology. Mesh topology can make such network have good fault-tolerant potential, and is suitable to provide broadband Internet access services.

1.1 Motivation

All wireless networks above may integrate together to form a hierarchical mobile network to provide users ubiquitous wireless access. Users may roam between various wireless networks. However, the most important problem is how to provide transparent Internet connectivity without user involvements.

Two major issues of providing transparent connectivity are address auto-configuration and seamless roaming capability. Since IP addresses represent relative logical locations of mobile hosts, mobile hosts need to change IP addresses when they roam into a new network. Address auto-configuration are required to help mobile hosts configure an address automatically without users involvements when they join in a new network. The other requirement is seamless roaming ability, which helps mobile hosts maintain connections if they switch to another network. Again, this should be achieved without disturbing users.

In addition to transparent Internet connectivity, information storage philosophy might also need some minor modifications under the environment of hierarchical mobile networks. In hierarchical mobile networks, users have high flexibility of movement and selecting mobile devices. This might cause information to be stored anywhere or in any device. It is important to information service providers to efficiently organize information for users.

1.2 Overview of proposed mechanisms

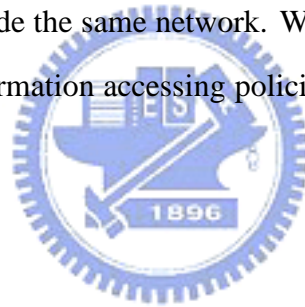
In this thesis, we proposed effective addressing and routing mechanisms in hierarchical mobile networks. The proposed addressing mechanism is originated from the canonical factorization theorem of positive integers. Each positive integer can be presented as a product of a unique sequence of prime numbers. Based on this characteristic of positive integer, proposed addressing mechanism can guarantee uniqueness of addresses assigned by different nodes.

The proposed routing mechanism utilize addressing results, which can generate a unique address allocation tree to help mobile hosts find routing paths. Since the address allocation tree is unique, each host can easily realize the logical position of itself and destination on the tree. This can enable each host to find routing paths without exchanging routing information or sending routing requests.

We also proposed a personal information hierarchy in this thesis. As mentioned earlier, information might be distributed all over the network while wireless accesses provides high flexibility and convenience to users. However, this may cause problems for users to manage their personal information. The proposed personal information hierarchy learns successful experience of memory hierarchy in computer systems to help users manage their personal information with both advantages of portability and storage space.

1.3 Synopsis

The remainder of this thesis is organized as follows. Chapter 2 describes the system architecture of hierarchical mobile networks. In Chapter 3, we present two proposed addressing allocation methods, DHCP relay method and DHCP proxy method. Chapter 4 and Chapter 5 describe internal and external routing mechanisms respectively for mobile hosts communicate with other hosts inside or outside the same network. We then describe the proposed personal information hierarchy and information accessing policies in Chapter 6. At last, we conclude the thesis in Chapter 7



Chapter 2

System Architecture

With the advance of embedded computing technologies, portable devices, such as laptops, Personal Digital Assistants (PDAs), and cellular phones, have been widely used. A portable device may even have several wireless interfaces, such as IEEE 802.11 Wireless LAN (WLAN), General Packet Radio Service (GPRS), Personal Handy-phone System (PHS), and/or Bluetooth. In this Chapter, we introduce some wireless networks and an integrated hierarchical mobile networks.

2.1 Mobile ad-hoc networks

Wireless communications are typically supported in two models: *infrastructure* and *ad hoc*, as illustrated in Figure 2.1. Among these two options, forming a *mobile ad hoc network (MANET)* is more flexible since it is independent of the availability of base stations. A MANET is a network consisting of a set of mobile hosts, which can roam around at their own will. Since no base stations are supported in such an environment, hosts may have to communicate with each other in a *multi-hop* manner. Applications of MANETs occur in situations like battlefields, disaster areas, and outdoor assemblies. Hence, intensive research has been dedicated to MANET [31, 42, 49, 61].

A working group called manet has been formed by the Internet Engineering Task Force (IETF) to study the related issues and stimulate research in MANET [50, 4, 54, 46]. The routing protocols in MANETs can be divided into two categories: *Reactive MANET Protocols (RMPs)* and *Proactive MANET Protocols (PMPs)*. RMPs are on-demand routing protocols, in which mobile hosts send routing requests only when necessary. PMPs are table-driven routing

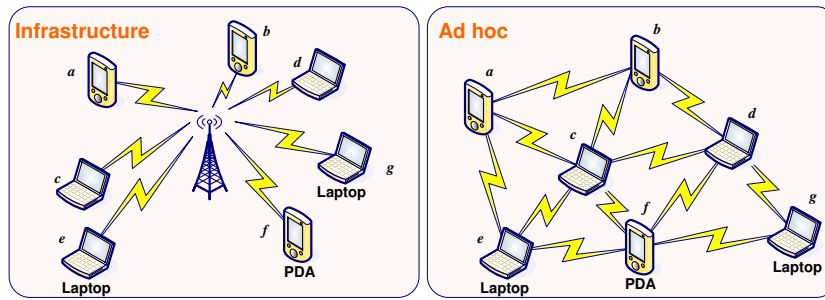


Figure 2.1: Two options of wireless networks: infrastructure and ad hoc.

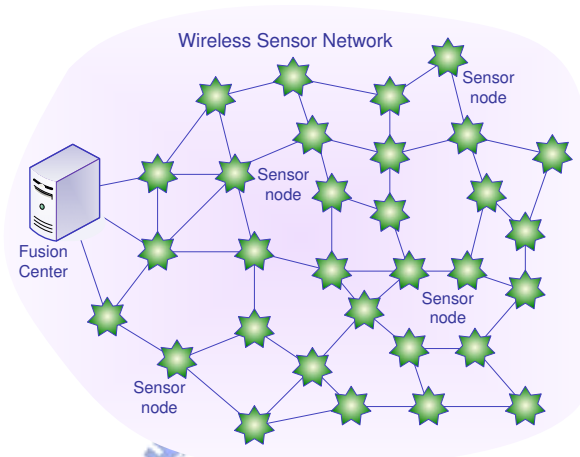


Figure 2.2: An example of wireless sensor network.

protocols, which make mobile hosts exchange routing information periodically and maintain routing paths to other hosts. Since MANET is a multi-hop wireless network and mobile hosts can move around freely, researchers need to design dynamic protocols with limited signal overhead.

2.2 Wireless sensor networks

A *wireless sensor network (WSN)* [44, 12, 8, 10, 13, 25, 33, 66] is a wireless network for observing some phenomenon such as temperature, pressure, or relative humidity. As shown in Figure 2.2, a WSN is a highly distributed networks consisting of a large number of small sensor nodes, which are densely deployed either inside the phenomenon or very close to it, and at least one data fusion center. WSNs like MANETs are multi-hop wireless networks. The main purpose of WSNs is to collect sensed data and send these to the fusion center. Instead of sending the raw data to the nodes responsible for the fusion, sensor nodes use their processing abilities to locally carry out simple computations and transmit only the required and partially

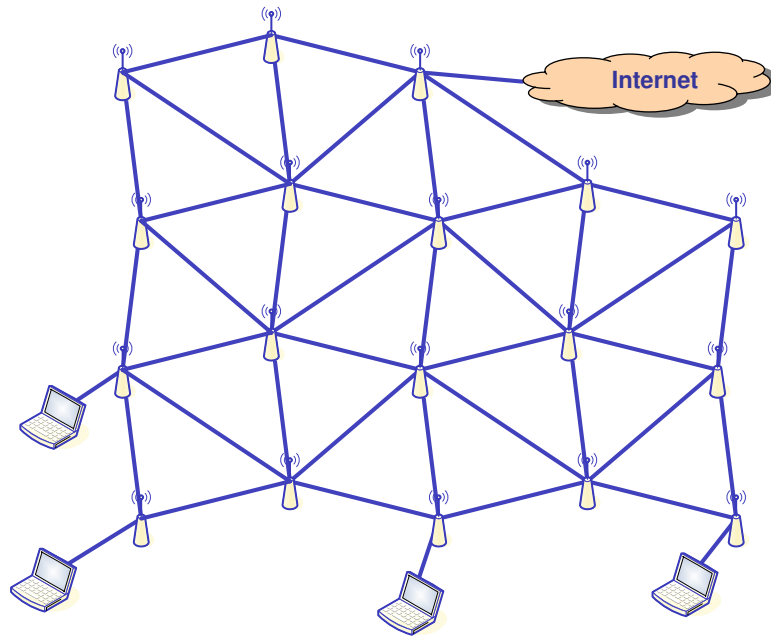


Figure 2.3: An example of wireless mesh network.

processed data. Through wireless network, fusion center will deal with the transferred data from sensors and report useful information to observer.

The power consumption of sensor nodes is always critical since it is almost impossible to recharge each sensor node. In order to save power, the transmission range of each sensor node is usually limited to several meters. As a result, WSN applies multi-hop forwarding if fusion center is far away from some sensors. Due to this limitations, there are lots of problems, like power consumption, scalability, reliability or fault tolerant problems, need to be solved in WSNs.

2.3 Wireless mesh networks

Although IEEE 802.11 WLAN wireless broadband networks has become very popular in recent years, the limited reach of signal propagation and high cost of installing and maintaining a wired network backhaul connection have limited WLAN network deployments to homes, offices, public hot spots (in coffee shops, airports, hotels, and other similar locations), and some wide-area hot zones. In addition, installing, managing, and scaling multiple hot spots is very difficult.

A wireless mesh network [26, 36, 35, 1] has potential to overcome these limitations to

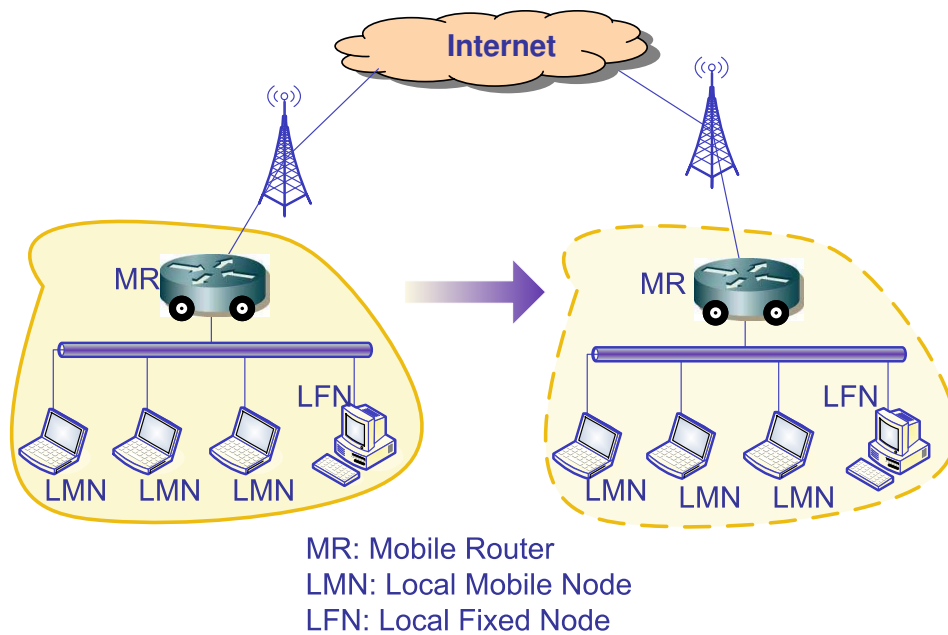


Figure 2.4: An example of network which is mobile (NEMO).

create truly unwired cities. As shown in Figure 2.3, backhaul routers in a wireless mesh network have multiple wireless links to other backhaul routers. These backhaul routers can be deployed without wiring. A mesh network could be defined as a network that employs one of two connection arrangements, full mesh topology or partial mesh topology. In the full mesh topology, each node is connected directly to each of the others. In the partial mesh topology, nodes are connected to only some, not all, of the other nodes.

A wireless mesh network is considered as a promising alternative for broadband Internet access. To provide reliable Internet access, there are still some problems need to be solved before it could be deployed in reality. These problems include efficiently QoS routing, fault tolerance, auto-configuration, ... etc.

2.4 Network mobility

Although lots of research has been done on host mobility supports that aim to provide continuous Internet connectivity to mobile users [5, 14], it is possible that an entire network may move as a unit and change its point of attachment to the Internet dynamically.

A network which is mobile (NEMO) [55, 40, 59, 57] consists of a *mobile router (MR)* and all its attached nodes, through either wired or wireless interfaces. The MR changes its point of

attachment to the Internet dynamically while it is moving. Similar to the IP addressing method used in fixed networks, all IP addresses of the nodes in the MONET have the same IP prefix as the MR does. Besides, the nodes attached to the NEMO may themselves be mobile with respect to the MR. Figure 2.4 gives an example of NEMOs that may possibly be deployed in a mass transportation like a train. All nodes, including MR, local mobile nodes (LMNs), and local fixed nodes (LFNs), in the train form a NEMO and move together. MR could be equipped with multi-tier wireless interfaces such as GPRS, Wireless LAN and/or Bluetooth. LMNs may attach to the NEMO with a low-power but high-bandwidth wireless interface such as wireless LAN. Both LMNs and LFNs may access the Internet through the MR. Furthermore, an LMN or LFN may itself be a router to form a hierarchy of NEMOs. There is another smaller scale NEMO, which is personal area network (PAN). A user may bring several mobile devices, and these devices can form a PAN and one of the devices can help all devices in a PAN access the Internet.

The concept of mobile networks could extend the reachability of hosts in NEMOs, and possibly help Internet Service Providers (ISPs) in providing seamless services to more users (with fixed or mobile devices). For example, a NEMO deployed in an airplane, a boat, a train or a bus could provide the Internet connectivity to the passengers with fixed or mobile devices such as desktops, laptops, pocket PCs, PDAs, or mobile phones. Furthermore, although LMNs or LFNs move with the NEMO, they are static relative to the NEMO. Therefore, they do not change their points of attachment and need not make any location update while the NEMO is moving. In addition, the MR of the NEMO can connect to the Internet with whatever external wireless interface that is appropriate. The local nodes of the NEMO need not be aware of the changes in the external wireless interfaces of the MR.

Although NEMOs can extend our accessibility to the Internet, some problems [56, 60] still remain to be solved before the NEMOs can be widely deployed.

2.5 Hierarchical mobile networks

In previous sections, we introduce several wireless networks with different topologies and purposes. Among these wireless networks, MANETs, WSNs, and wireless mesh networks are

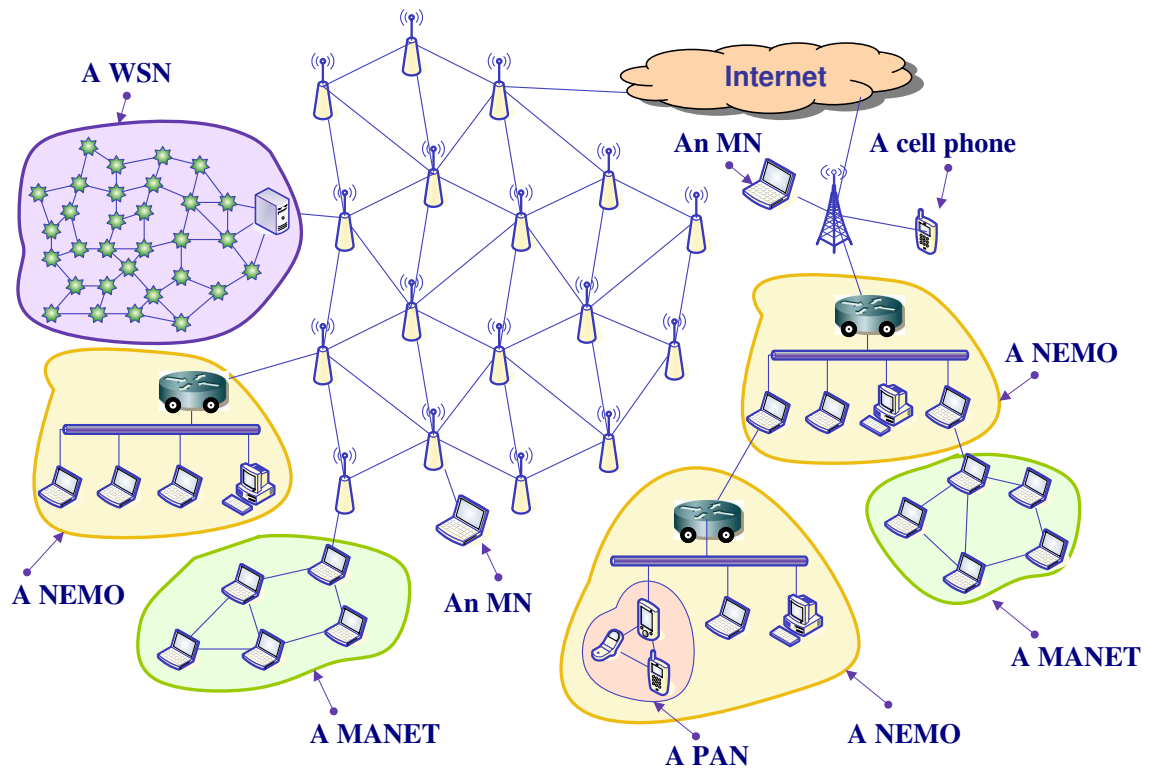


Figure 2.5: System architecture of hierarchical mobile network.

multi-hop wireless networks and NEMOs are a single-hop networks. These networks might together form a hierarchical network. As shown in Figure 2.5, a mobile node (MN) could attach to an infrastructure directly or attach to a wireless mesh network, MANET, or NEMO. A MANET, NEMO, or PAN could also attach to the Internet, a wireless mesh network, or even another NEMO. When a network attaches to another network, there must be a node acting as bridge of two different networks. We define such nodes as *gateways*. For example, MRs are gateways of NEMOs. If there are more than one gateway in a MANET, we define *sub-MANET* to represent a sub-network consisting of a gateway and all mobile nodes attaching to such gateway. Similarly, there might be *sub-NEMOs* or *sub-WSNs*.

In a hierarchical mobile network, a mobile node or a NEMO can freely roam between different networks. This could provide high flexibility to mobile hosts, but it also causes problems in network management to support seamless roaming capability to mobile hosts. In the following chapters, we propose some network and information management mechanisms in hierarchical mobile networks.

Chapter 3

Addressing Mechanisms for Hierarchical Mobile Networks

We know that a mobile host needs to set an IP address before it starts to communicate with other hosts. Since prefix of an IP address represents logical position of a device, each network usually manages their addresses locally. Traditionally, a network can manage their addresses statically or dynamically. In static address management, users need to have an address in advance and configure the address manually. In dynamic address management, a network can apply some dynamic address allocation methods, such as Dynamic Host Configuration Protocol (DHCP) [45] to assign addresses.

In our architecture of hierarchical mobile networks, mobile hosts need to acquire an address whenever they roam into a new network. We can divide all networks in our architecture into two categories: single-hop and multi-hop networks. MANETs, WSNs, and wireless mesh networks are multi-hop networks and NEMOs belong to single-hop networks. It is impossible for mobile hosts to configure addresses manually in hierarchical mobile networks, so we need to find proper address allocation methods for each network. DHCP involves three rounds of broadcasting, which may cause huge signal overhead or even broadcast storm problem [51] in a multi-hop environment. Therefore, single-hop networks like NEMO in our architecture can apply DHCP directly, and we propose efficient addressing allocation mechanisms in multi-hop environments in this Chapter.

We will describe prior researches on this problem in the following section 3.1. We then propose two addressing allocation mechanisms for multi-hop environments in this Chapter. The first one is DHCP relay method in Section 3.2, which can reduce signal overhead. The

second one is DHCP proxy method in Section 3.3 to reduce both signal overhead and latency of address acquiring process.

3.1 Previous works

Several dynamic address allocation schemes have previously been proposed for MANETs. According to the prior research [53], they can be classified into three categories:

3.1.1 Best effort allocation

Schemes in this category can not guarantee address uniqueness. Prophet scheme [21] proposes a complex address generation function for each host to generate a sequence of addresses to be assigned to new coming neighbors. The proposed function tries to use huge address space to degrade the percentage of generating a same address from two nodes. When a new node joins a network, it can simply acquire an address from its neighbors. However, the proposed function in Prophet method still can not guarantee the uniqueness of addresses assigned by different nodes. In other words, there still exists probability that two nodes can generate a same address. Therefore, even with a large address space, prophet scheme may still needs some mechanisms, such as Duplicate Address Detection (DAD) or weakDAD [37], to resolve address conflicts. DAD will cause broadcast storm problem and weakDAD will introduce extra packet overhead by adding MAC address into IP headers for all data packets.

3.1.2 Centralized allocation

A centralized server is deployed to manage all addresses in this category of address allocation schemes. DHCP [45] is a typical example, but it needs broadcasting for both server discovery and DAD. ODACP [53] attempts to reduce broadcasting by having the server broadcast advertisement periodically so a new host can directly register its addresses to the server. A longer advertisement interval does help to reduce the overhead of broadcasting, but it also results in longer latencies for hosts to obtain addresses.

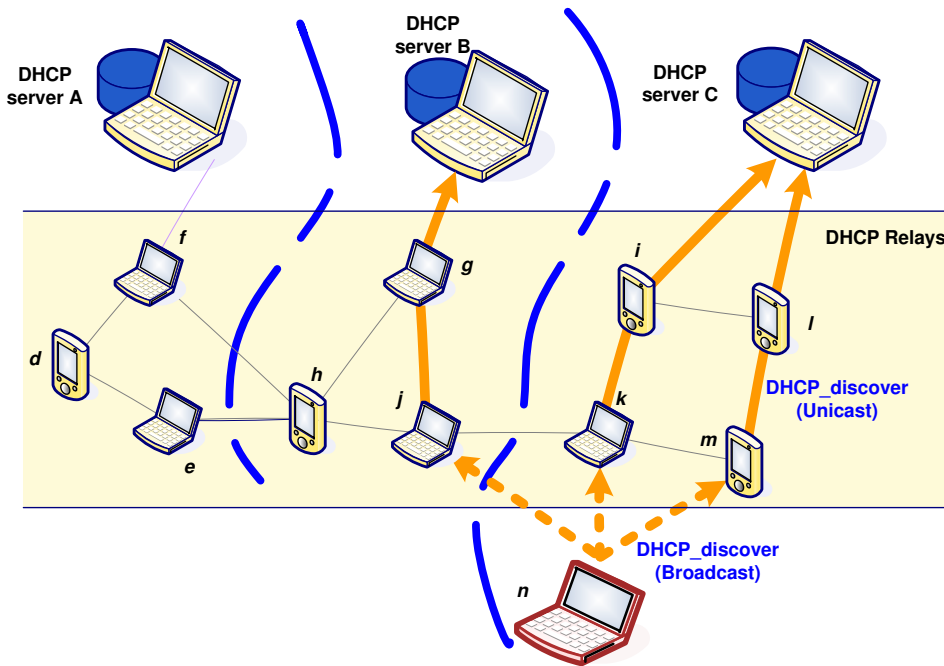


Figure 3.1: Address resolution and DHCP operations for host n to join the MANET.

3.1.3 Decentralized allocation

A host could acquire an address by itself or from a neighbor and then performs DAD to ensure the uniqueness of the address. In AAA [9], hosts randomly select an address in the range of 169.254/16. In MANETconf [34], each host stores all addresses used in the MANET, and a new coming host acquires an address from one of its neighbors. The neighbor then broadcasts a query, on behalf of the new host, for DAD throughout the network. Mohsin [32] employs a buddy system for address allocation, but it is difficult to manage address blocks among all MANET hosts.

3.2 DHCP relay method

The first address allocation method we propose is DHCP relay method [23]. As shown in Figure 3.1, we install at least one DHCP server in each MANET. If there are more than one DHCP server in a MANET, we define a *sub-MANET* consisting of a DHCP server and all nodes with addresses assigned by such DHCP server. To avoid confusion, we assign an exclusive section of IP addresses to each DHCP server. Note that we also allow a host to use its old IP address after roaming into a new sub-MANET.

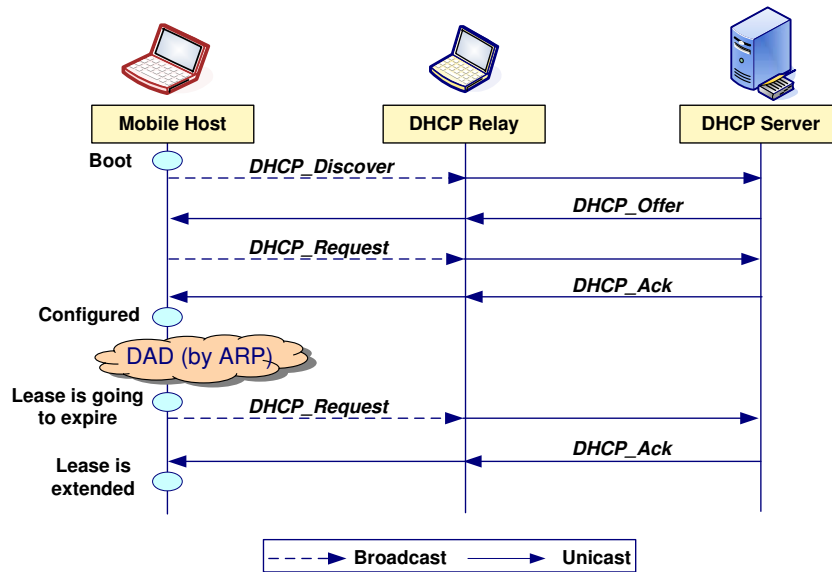


Figure 3.2: The message flow of DHCP in our architecture.

Taking Figure 3.1 for example, when a new mobile host n joins the MANET, it first broadcasts a *DHCP_discover*. Three nodes, j , k , and m , belonging to two sub-MANETs receive the request and help forwarding the request to their DHCP servers. To avoid the *broadcast storm* problem [51], the forwarding is done by unicast. This can be achieved by configuring each internal mobile host as a DHCP relay.

Figure 3.2 illustrates the related DHCP message flows. The *DHCP_Discover* is forwarded by the DHCP relay to the DHCP server via DSDV routing. The DHCP server then replies a *DHCP_Offer* by including an available IP address. Note that the host may receive multiple offers from several servers. So the host will broadcast a *DHCP_Request* to notify all DHCP servers the IP address that it selects. Again, the notification will be supported by unicast. The selected server then replies a *DHCP_Ack* if the IP is still available. Afterward, the *Duplicate Address Detection (DAD)* procedure will be executed to ensure that no other host is using the same IP address. There is also a parameter, called *lease*, after which the mobile host has to renew its temporary IP address with the same DHCP server. After obtaining an IP address, the host will turn itself into a DHCP relay of the DHCP server that it selects.

By configuring hosts except DHCP servers as DHCP relays can make messages be sent by one-hop broadcasting instead of whole-network broadcasting. Therefore, the proposed DHCP relay method can reduce signal overhead. Unfortunately, a host still needs to wait until DHCP

server sending back the response. In the next section, we will propose a DHCP proxy method to further reduce latency.

3.3 DHCP proxy method: prime DHCP

In this section, we propose a *Prime DHCP* scheme for address allocation without broadcasting in the whole MANET during the address allocation process. In the proposed prime DHCP, each host serves as a DHCP proxy that can assign addresses to new hosts by running a proposed *Prime Numbering Address Allocation (PNAA)* algorithm individually to compute unique addresses for address allocation. The use of DHCP proxies and the PNAA together eliminate the need for broadcasting in the whole MANET.

As mentioned in the Section 3.1, almost all previous address allocation schemes for MANETs rely on broadcasting for server discovery or DAD. We propose Prime DHCP to achieve these two functionalities without broadcasting. Prime DHCP configures each host as a *DHCP proxy*, so all hosts are eligible to assign addresses and a new host can acquire an address simply from its neighbors. Besides, each DHCP proxy runs PNAA individually to compute unique addresses for address allocation so that DAD is not required in prime DHCP.

3.3.1 Address allocation tree

In order to eliminate the necessity of DAD, we propose a PNAA algorithm to guarantee the uniqueness of addresses. PNAA is originated from the canonical factorization theorem of positive integers, that is, every positive integer can be written as a product of prime numbers in a unique way.

Before describing proposed mechanism, we first define prime factorization sequence of an integer in Definition 1. According to the canonical factorization theorem [17], each integer can be expressed as a production of prime numbers. For example, integer 12 is equal to $2*2*3$, and thus $pfSeq(12) = (2, 2, 3)$. We prove the prime factorization sequence of an integer is unique in Lemma 1.

Definition 1. *The prime factorization sequence of an integer $n > 1$, written $pfSeq(n)$, is an ascending ordered set of all prime factors of n , where the product of all elements in the set is*

equal to n .

$$pfSeq(n) = (p_0, p_1, p_2, \dots, p_m) \text{ where } p_i \leq p_j \forall i < j \text{ and } \prod_{i=0}^m p_i = n.$$

Lemma 1. *The prime factorization sequence of each integer $n > 1$, $pfSeq(n)$, is unique.*

Proof. We prove this lemma by mathematical induction.

Basis: $pfSeq(2) = (2)$, which is unique.

Hypothesis: assume $pfSeq(n)$ is unique $\forall n \leq k$

Induction: $n = k + 1$, assume $pfSeq(n)$ is not unique.

$$pfSeq(n) = (p_0, p_1, \dots, p_l) = (q_0, q_1, \dots, q_m)$$

We can prove $pfSeq(n)$ is unique by showing $p_0 = q_0$ since $pfSeq(n/p_0)$ is unique according to the hypothesis.

$$\begin{aligned} \therefore n &= p_0 \cdot p_1 \cdot \dots \cdot p_l = q_0 \cdot q_1 \cdot \dots \cdot q_m \\ \Rightarrow p_0 &| (q_0 \cdot q_1 \cdot \dots \cdot q_m) \\ \Rightarrow p_0 &| q_j \text{ for some } j \\ \Rightarrow p_0 &= q_j (\because p_0 \text{ and } q_j \text{ both are prime numbers}) \\ \Rightarrow p_0 &\geq q_0 (\because q_j \geq q_0) \end{aligned}$$

Similarly, $p_0 \leq q_0$

$$\Rightarrow p_0 = q_0$$

$\therefore pfSeq(n)$ is unique when $n = k + 1$.

By mathematical induction, the prime factorization sequence of each integer is unique. \square

Figure 3.3 gives an example of addresses each DHCP proxy can assign. The first node in the network is *root proxy*, and it configures its address as 1. The root proxy allocates all prime numbers, in ascending order, to new nodes attached to it. For a non-root DHCP proxy with address n and $pfSeq(n) = (p_0, p_1, \dots, p_m)$, it can assign the address equal to its own address multiplied by a prime number, starting from its largest prime factor p_m . Each node can also identify the address of its parent simply by dividing its own address by the largest prime factor

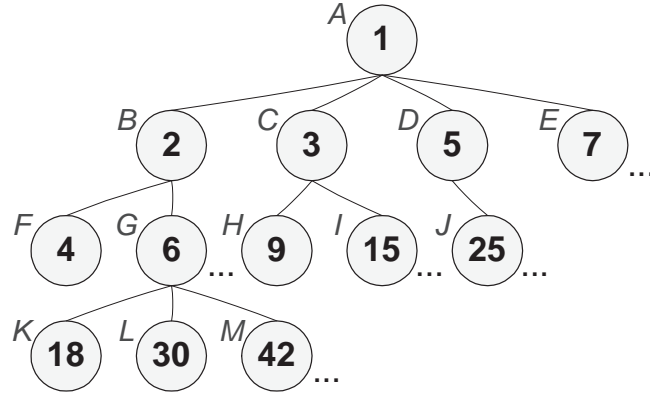


Figure 3.3: An example of address allocation tree.

of its own address. Take node G with address 6 for example, the largest prime factor of 6 is 3 and the sequence of addresses node G can assign is, $6 * 3$, $6 * 5$, $6 * 7$, and so on up to the largest address bounded by the address space. Node M in Figure 1 has the address 42, and the address of its parent is $42/7 = 6$.

Because the prime factorization sequence of each integer or address is unique, we can prove in Theorem 1 that no two proxies can generate the same address by running the proposed PNAA algorithm. Therefore, DAD is not necessary during the process of address resolution. Besides, in Theorem 2, we also prove that each integer or address is assignable, so there is no hollow address and we can utilize all addresses efficiently.

Theorem 1. *No two nodes will assign the same address in PNAA.*

Proof.

Given an integer n , $pfSeq(n) = (p_0, p_1, p_2, \dots, p_m)$

According to PNAA, the address assignment path of integer n is

$(1, \prod_{j=0}^0 p_j, \prod_{j=0}^1 p_j, \prod_{j=0}^2 p_j, \prod_{j=0}^m p_j)$

$\because pfSeq(n)$ is unique (proved in Lemma 1)

\therefore the address assignment path of integer n is also unique,

and thus no two nodes can assign the same address.

□

Theorem 2. *Each address is assignable in PNAA.*

Proof. We prove this theorem by mathematical induction.

Basis: Address 1 is assignable because it is the address of the first node (root) in the network.

Hypothesis: For all positive integer $n \leq k$, n is assignable.

Induction: For $n = k + 1$, and $pfSeq(k + 1) = (p_0, p_1, \dots, p_m)$

By Lemma 1, $pfSeq(k + 1)$ is unique.

case 1. $k + 1$ is a prime number: $k + 1$ is assigned by the root.

case 2. $k + 1$ is not a prime number:

$\because (k + 1)/p_m$ is less than k

$\therefore (k + 1)/p_m$ is assignable according to the hypothesis

$k + 1$ is assigned by the node with address $(k + 1)/p_m$

$\therefore n = k + 1$ is also assignable.

By mathematical induction, all positive integer n is assignable. □

3.3.2 Address allocation procedures

After being sure that each node can assign disjoint addresses by running PNAA, we describe operations of proposed prime-based self-configured addressing mechanism. First, each DHCP proxy maintains a sequence of prime numbers and its own *allocation status*, a pointer to the prime number multiplied for the last assigned address. When a new node joins a network, Figure 3.4 illustrates the message flows of how the new coming node acquires an address. First of all, the new node issues a *DHCP_Discover* broadcast message as a normal DHCP client does and starts a timer. When the neighbor nodes, DHCP proxies 1 and 2, receive the message, they runs PNAA algorithm to generate an address, and then encapsulates the address in a *DHCP_Offer* message. Before the timer expires, the new node gathers all *DHCP_Offer* messages with available addresses, and it chooses the smallest address to prevent the address allocation tree from growing too fast. The choice is set in *DHCP_Request* broadcast message to inform all its neighbors. Finally, the chosen proxy 1 updates its address allocation status and then sends a *DHCP_Ack* to the new node for confirmation. Consequently, instead of

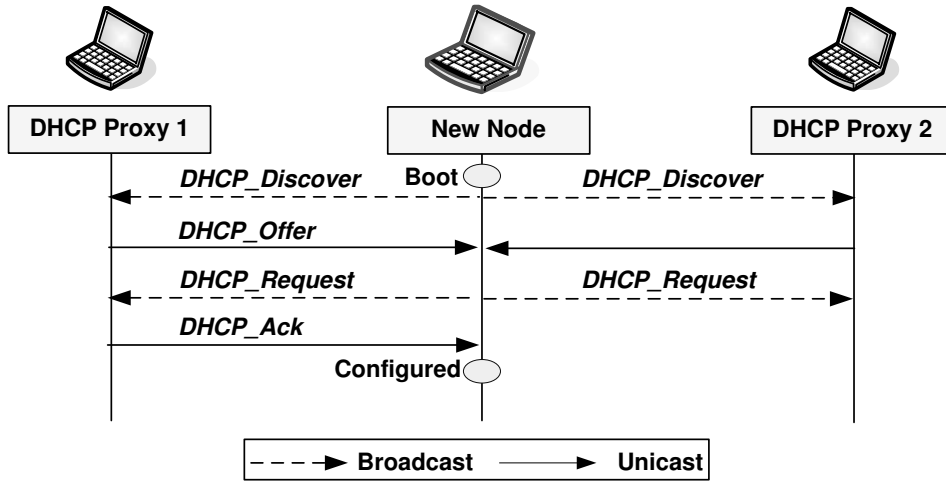


Figure 3.4: Message flows of the address allocation procedure of a new coming node.

whole-MANET broadcasts, *DHCP_Offer* and *DHCP_Request* incur just a single-hop broadcast, which is necessary anyway in the wireless environment.

We use Figure 3.5 as an example MANET to illustrate the operations of Prime DHCP. It should be noted that Figure 3.3 is just a logical tree of the address allocation. Once a host obtains an address, it can move freely and continuously use the address as long as it is still in this network. Figure 3.5 is a possible physical topology of MANET hosts in Figure 3.3. Without loss of generality, we assume that the address space of this MANET is 128 in the following discussion. Furthermore, the messages can be forwarded by any proactive routing protocols for MANETs.

When a new mobile host N joins a MANET, it issues a *DHCP_Discover* broadcast message as a normal DHCP client does. When the neighbor hosts, L , B , and F , receive the message, they start serving as DHCP proxies of the host N . Rather than forwarding the broadcast message further, each DHCP proxy runs PNAA algorithm to generate an address, and then encapsulates the address in a *DHCP_Offer* message. Note that proxy L relays the *DHCP_Discover* to its parent proxy G for help because its minimum assignable address is 150, which exceeds the address space 128. Therefore, host N receives three *DHCP_Offer* messages offering addresses 66, 10, and 8, respectively, from DHCP proxies G , B , and F . Host N chooses the smallest address 8 to prevent the tree from growing too fast, and broadcasts its choice in a *DHCP_Request*. Again, its neighbors do not flood the message further, but proxy L relays the message to its parent proxy G . Finally, the chosen proxy F updates its address alloca-

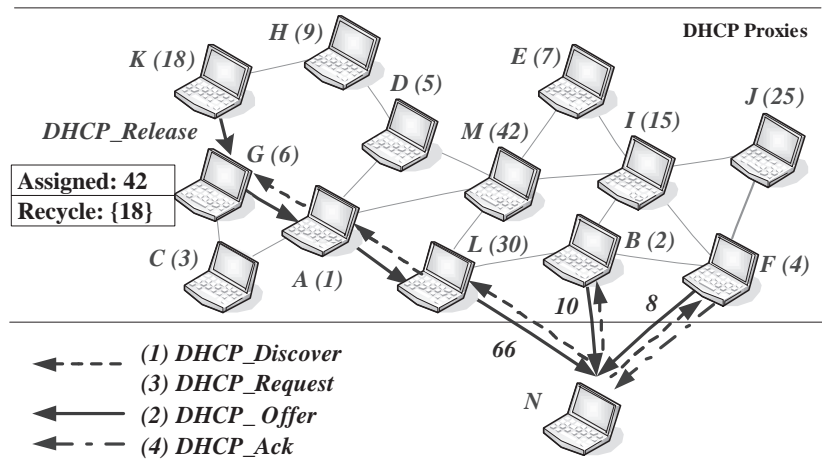


Figure 3.5: An Example of MANET topology with a new coming host.

tion status and then sends a *DHCP_Ack* to host *N* for confirmation. Consequently, instead of whole-MANET broadcasts, *DHCP_Offer* and *DHCP_Request* incur just a single-hop broadcast, which is necessary anyway in the wireless environment, and possibly seldom multi-hop unicasts.

Figure 3.6 gives an example of address allocation results for a 3x3 mesh network. We assume the order of nodes joining the network is from node A to node I. In the first step (a), node A is the only node in the network, so it configures its own address as 1. Two nodes B and C then attach to node A and obtain address 2 and 3 respectively as shown in step (b). In step (c), node E receives two addresses 6 and 9 from node B and C respectively, and it chooses the smaller one as its address. Similarly in step (d), node G also chooses the smaller address, 8 among 8 and 18. In steps (e) and (f), nodes H and I choose the smaller addresses 18 and 16 from node E and G, respectively. Figure 3.7 is the address allocation tree of Figure 3.6. The solid lines represent branches of the address allocation tree, whereas the dotted lines are the links not on the address allocation paths.

In order to avoid the address leak problem, a host should depart *gracefully* by informing its parent of leaving, and each host maintains a recycle list to record the allocation statuses for its departed children in its allocation status. For example, proxy *K* in Figure 3.5 is leaving, so it sends a *DHCP_Release* message to its parent proxy *G*. When proxy *G* receives a *DHCP_Discover* from a new node, it will first give out the lowest recycled address and inform the new node the allocation status associated with the offered address. If the root is about to

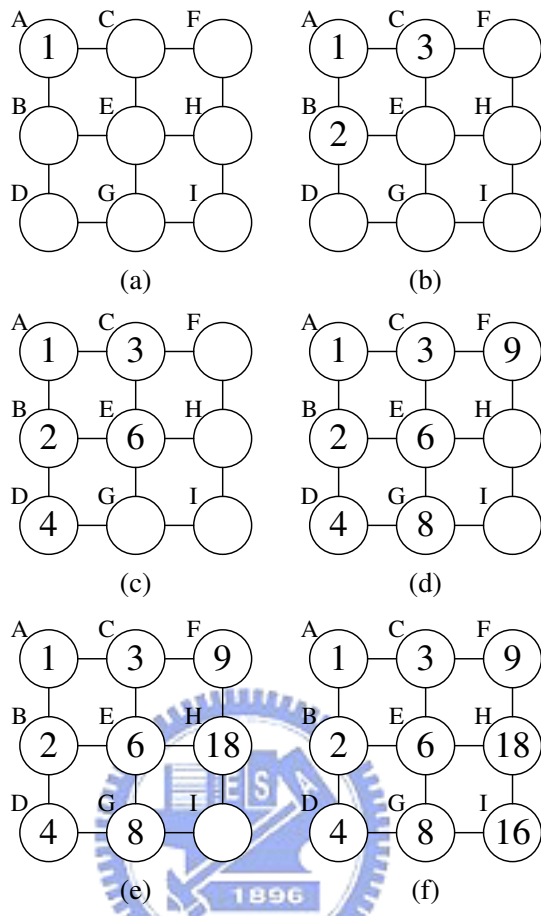


Figure 3.6: An example of address allocation result.

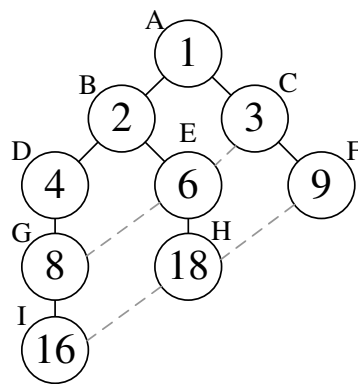


Figure 3.7: The address allocation tree.

leave, it informs its greatest descendent to be the new root proxy. However, it is likely that a host may leave gracelessly, packets may be lost, address may be limited, or MANETs may merge or split. In the next subsection, we will explain how the prime DHCP handles these exceptions.

3.3.3 Exception handling

Most message losses, except *DHCP_Release* that can be treated as a graceless departure, can be recovered by having the sender set a timer and resend the messages when the timer expires. For a host left gracelessly, its address may become not usable. In order to reclaim leaked addresses, the root proxy periodically broadcasts a *DHCP_Recycle* message to ask all hosts report their address allocation statuses, including addresses they have given out or recycled. By gathering statuses from existing hosts, the root proxy can reconfigure the *address allocation tree*, and inform each proxy of updated allocation status. The *DHCP_Recycle* broadcast message could be piggybacked in the messages of MANET routing protocol, DSDV [43] for example, so no additional overhead will be introduced by the address recycling.

Network merging and partition can also be detected by periodically recycle process. When two MANETs merge, a root can receive allocation statuses from nodes in the other network, detect address conflicts, and ask one of the two conflicting nodes to inquire a new address. On the other hand, although network partition will not result in address conflicts, the split MANET needs a new root for address recycle. If a node misses several recycle messages, it may claim to be the new root by sending a *DHCP_Recycle* after a backoff time reversely proportional to its address. This can make the node with the largest address the new root to increase address utilization.

When the address space is limited, a node might gain an address exceeding the range of addresses by running PNAA while receiving an address request. In such case, the node relays the request to its parent node to obtain a legal address. Assuming the address space of the network in Figure 3.6 is limited to 20, we illustrate the last two steps of address allocation results in Figure 3.8.

If a DHCP proxy generates an address greater than the address space, it relays the request message to its parent node to retrieve an address. As shown in step (e), node H acquires

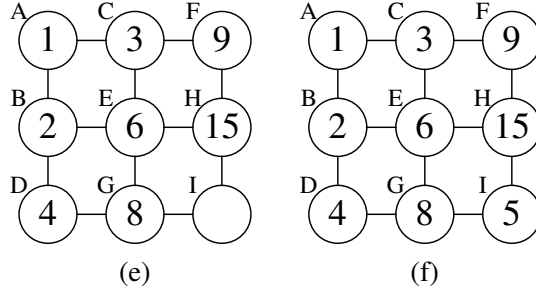


Figure 3.8: The last two steps of Figure 3.6 when address space=20.

Table 3.1: Qualitative analysis of address allocation mechanisms

	DHCP	MANETconf	ODACP	Prophet	PrimeDHCP
Uniqueness	Yes	Yes	Yes	No	Yes
Signal overhead	$O(4l)$	$O(2l)$	$O(2l)$	$O(n/2)$	$O(n/2)$
Latency	$O(4td)$	$O(2td)$	$O(2td)$	$O(2t)$	$O(2t)$
Complexity	Low	High	Low	High	Low

address from its neighbor node E and F. The next address node F can assign is 27, which is larger than the address space, so it relays the request to its parent node C and obtains an address 15. Since 15 is smaller than 18, which is assigned by node E, node H chooses 15 as its address. In the last step, node I similarly obtains address 5 from neighbor node H.

3.3.4 Performance evaluation

Table 3.1 shows the qualitative analysis of the proposed prime DHCP and other address allocation mechanisms. Assume the numbers of hosts and links are n and l , respectively; the network diameter is d ; and the average one-hop latency is t . First, Prophet is the only method that can not guarantee the uniqueness of addresses. Second, DHCP needs to perform server discovery and DAD, so at least $4l$ hosts need to process signal packets and the latency is $4*t*d$. MANETconf and ODACP need to perform DAD and server advertisement respectively, and thus the overhead and latency are $2l$ and $2*t*d$. Prophet and Prime DHCP both send requests to neighbors only, so the overhead is the average degree ($n/2$) of each node in the network and the latency is $2t$, assuming that the address space is sufficient. At last, MANETconf and Prophet scheme involve more complicated computation in address allocation.

Since the behaviors of address assignment in Prime DHCP and Prophet scheme are similar, the latency and overhead of Prime DHCP can be referred to [21]. Here we focus the analysis on

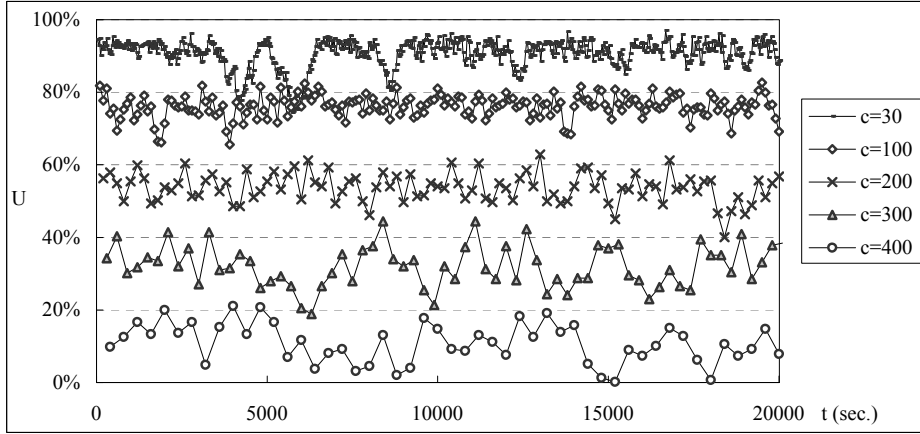


Figure 3.9: The effect of recycle period on the address utilization.

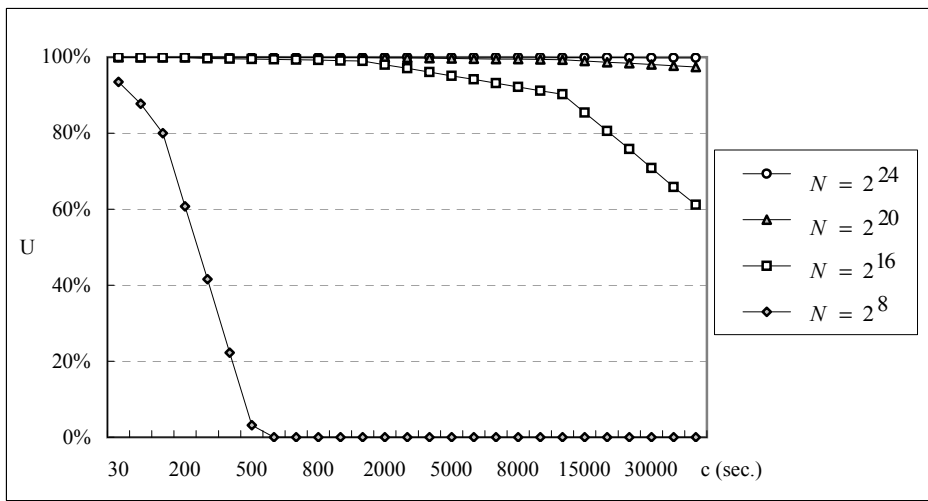


Figure 3.10: The effect of address space on the address utilization.

how recycle period affects the address utilization, i.e., the percentage of the effective addresses that Prime DHCP has given out or can possibly assign currently. Suppose the arrival and departure of hosts follow a Poisson distribution with the mean rates λ and μ , respectively. The address utilization U_k before the k th recycle can be represented as Equation (1), where N is the size of address space; $\tilde{n}(C_k)$ is the number of hosts in the MANET at the time of k th recycle; and is the average number of descendants of a host.

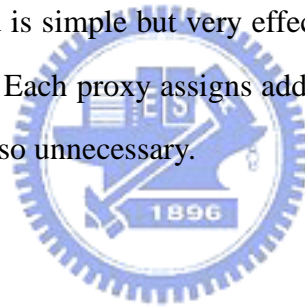
$$U_k = 1 - \frac{\tilde{n}(C_k) * \mu * \text{graceless percentage} * \bar{N}_{subtree}}{N}. \quad (3.1)$$

Figure 3.9 shows the address utilization U_k at different time instances for different recycle periods (c), where $N = 256$, $\lambda = 0.8$ and $\mu = 0.9$. As shown in the figure, when the recycle period becomes shorter, the recycle process will be performed more frequently but utilization

will be better. We can also observe from Figure 3.10 that although the utilization decreases as the recycle period increases, the speed of descent is slow if the address space is sufficiently large. Therefore, address recycling introduces only slight overhead if the MANET has an address space greater than class B, and this could be achieved by configuring the MANET as a private network.

3.4 Summary

We apply DHCP directly for address allocation in single-hop networks. In multi-hop networks, we propose two methods to reduce signal overhead and latency of DHCP. The DHCP relay method configures all hosts except DHCP server as DHCP relay, so all DHCP signal message will be relayed to DHCP server directly without be flooded all over the whole network. The DHCP proxy method we proposed is Prime DHCP to make each host as a DHCP proxy. The proposed Prime DHCP method is simple but very effective in terms of both signal overhead and address allocation latency. Each proxy assigns addresses according to the prime number tree, so DAD broadcasting is also unnecessary.




Chapter 4

Internal Routing Mechanisms for Hierarchical Mobile Networks

After solving address allocation problem, we next need to solve the problem of routing in hierarchical mobile networks. We first need to make sure that each mobile host can communicate with other hosts in the same network, which is internal routing.

4.1 Motivation



There are four different wireless networks in our hierarchical mobile networks. Among these four networks, NEMO is single-hop network. Mobile hosts in a NEMO can just send packets to MR and then to other hosts like original local routing in Local Area Networks. MANETs have flexible network topologies, and there are already lots of routing protocols proposed for MANETs. The left two wireless networks, wireless sensor networks and wireless mesh network, are both multi-hop networks with almost fixed network topologies. They do not need complicated mechanisms designed for MANETs since they have fixed topology, but it is also not proper for them to apply routing mechanisms in wired environments due to huge signal overhead caused by multi-hop behaviors. Therefore, we propose a prime-based self-configured routing protocol in Section 4.2 to solve internal routing problems for multi-hop wireless networks with fixed topologies.

4.2 Prime-based self-configured routing protocol

In this section, we propose a prime-based self-configured routing protocol, which is a routing protocol applied in a local network to solve internal routing problem. This protocol take advan-

tage of address results of PNAA described in Chapter 3. By employing PNAA, addresses of all nodes can form a unique addressing allocation tree which can represent location relationship of all nodes. Each node can easily be aware of locations of all nodes by their addresses, and this they can find routing paths by themselves without either periodically exchanging routing information or asking other nodes when necessary.

Before describing the proposed routing protocol, we define *Sequence GCD* of two integers in the Definition 2 first. Nodes can find their routing path by computing the sequence GCD of source and destination addresses.

In the following subsections, we will present how nodes find routing paths by just computing sequence GCD of the addresses of itself and the destination node. We proposed stateless and stateful routing protocols. In the stateless routing protocol, all nodes do not need to record anything, but the routing path may be longer. The stateful routing protocol can help nodes to find a shorter path but nodes need to record the addresses of its neighbors.

Definition 2. Assuming $pfSeq(a) = (p_0, p_1, p_2, \dots, p_m)$ and $pfSeq(b) = (q_0, q_1, q_2, \dots, q_n)$, we define $seqGCD(a, b)$ as the sequence GCD of integers a and b . $seqGCD(a, b) = \prod_{r_0}^{r_j}$ is the product of all elements in the longest common prefix sequence $y = (r_0, r_1, r_2, \dots, r_j)$ of $pfSeq(a)$ and $pfSeq(b)$. Which means j is the largest number that makes $r_i = p_i = q_i$ for all $0 \leq i \leq j$. If there is no common prefix of $pfSeq(a)$ and $pfSeq(b)$, $seqGCD(a, b)$ is equal to 1.

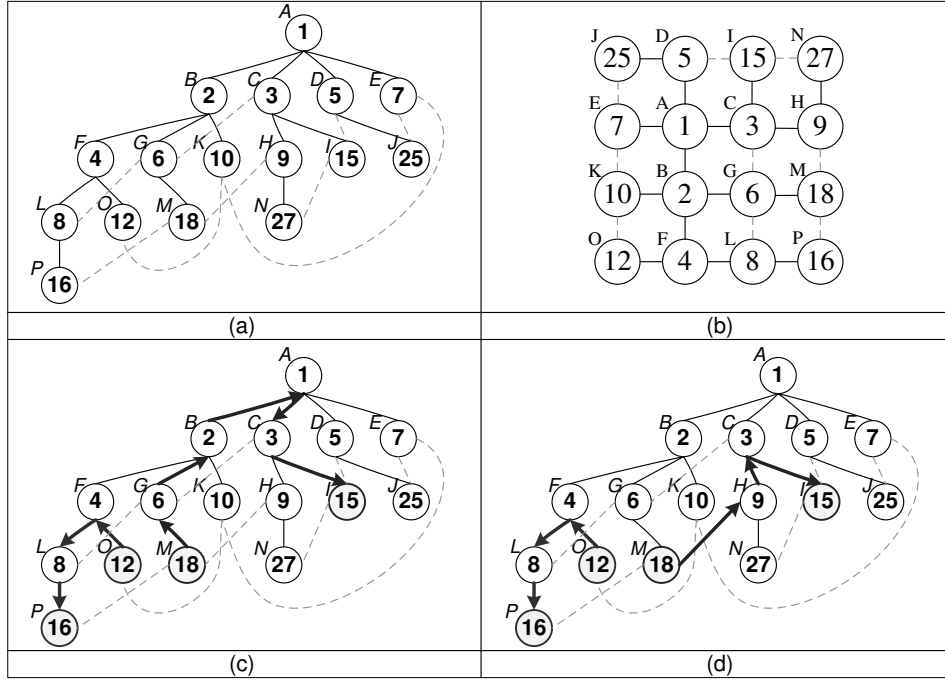


Figure 4.1: An example of setting routing path: (a) Address allocation tree; (b) Mesh topology; (c) Routing paths while applying stateless routing protocol; (d) Routing paths while apply stateful routing protocol.

E.g.,

$$seqGCD(16, 12) = 2 \times 2 = 4$$

$$\therefore pfSeq(16) = (2, 2, 2, 2)$$

$$pfSeq(12) = (2, 2, 3)$$

the longest common prefix of $pfSeq(16)$ and $pfSeq(12)$ is $(2, 2)$.

$$seqGCD(18, 15) = 1$$

$$\therefore pfSeq(18) = (2, 3, 3)$$

$$pfSeq(15) = (3, 5)$$

the longest common prefix of $pfSeq(18)$ and $pfSeq(5)$ is \emptyset .

4.2.1 Stateless prime-based self-configured routing protocol

In the stateless routing protocol, all nodes can configure routing paths without either exchanging routing information or recording any information. We provide two alternatives: source

```

1  x=seqGCD(src, dst)
2  // setting up the routing path from src to x
3  while(src!=x){
4      parent = src/(the largest prime factor of src)
5      add parent into the routing path
6      src=parent
7  }
8  // setting up the routing path from x to dst
9  int primeFactors[]
10 primeFactors = pfSeq(dst/x)
11 for(int k=0; k< number of elements in primeFactors; k++){
12     child =x
13     for(int i=0; i<k; i++){
14         child = child*primeFactor[i]
15     }
16     add child into the routing path
17 }

```

Figure 4.2: The procedure of setting routing path in the source routing method of stateless routing protocol.

routing method and hop-by-hop routing method. If source routing method is applied, the source node is responsible for setting the whole routing path, otherwise each node finds the next hop by itself to send packets.

We take Figure 4.1 for example. Figure 4.1 (a) is the address allocation tree of a mesh network in Figure 4.1 (b). In the stateless routing protocol, nodes purely proceed along branches of the address allocation tree to obtain the routing path. Figure 4.1 (c) illustrates two routing paths from node O to node P and from node M to node I. We can observe that the routing path can be divided into two segments: from the source node to the least common ancestor and then to the destination node. We will describe the detail procedures of two alternatives of stateless routing algorithms in the subsections below.

Source routing method

Figure 4.2 is the procedure of how a source node finds the whole routing path to the destination node, where *src* is the address of the source node and *dst* is the address of the destination node. The source node first finds the least common ancestor by computing $seqGCD(src, dst)$. Then it builds the first segment of routing path by recursively adding parent nodes until reaching the common ancestor as shown in the line 3 to 7 in Figure 4.2. For the second segment of the routing path, the source node needs to choose right child nodes. We know that the position of each node in the address allocation tree is determined by its address. Assuming $pfSeq(dst) = (p_0, p_1, p_2, \dots, p_m)$, we can guarantee that the address of the common ancestor

```

1  x=seqGCD(cur, dst);
2  if(x<cur){
3    parent = cur/(the largest prime factor of cur)
4    send packet to parent
5  }else if(x==cur){
6    child=x*(the smallest prime factor of (dst/x) )
7    send packet to child
8  }

```

Figure 4.3: The procedure of setting routing path in the hop-by-hop routing method of stateless routing protocol.

is $\prod_{i=0}^{i=j} p_i$ for some $0 \leq j \leq m$, and addresses of nodes along the path from the common ancestor to the destination are $(\prod_{i=0}^{i=j+1} p_i, \prod_{i=0}^{i=j+2} p_i, \dots, \prod_{i=0}^{i=m} p_i)$. Therefore, the source node can recursively add these nodes into the routing path as shown in the line 8 to 17 in Figure 4.2.

Taking Figure 4.1 for instance, node M(18) wants to set the routing path to node I(15). Node M(18) first computes $seqGCD(18, 15)$ and is aware that their common ancestor is node A(1). It starts to recursively add addresses of parent nodes, 6, 2, and 1, into its routing path. While reaching the common ancestor, node M(18) factorizes the quotient of 15 by 1 and obtains $pfSeq(15/1) = (3, 5)$. It then recursively adds $1 * 3$ and $1 * 3 * 5$ into the routing path. Consequently, it gets the whole routing path (6, 2, 1, 3, 15) to node I(15).

Hop-by-hop routing

Besides source routing method, we also propose another alternative, hop-by-hop routing, to enable each node to compute the next hop by itself. Although this method might increase the computation load of each node, it can decrease the header overhead introduced by source routing. Figure 4.3 is the pseudo code of how a current node with address cur finds the next hop for a destination having address dst .

First of all, the current node also computes $seqGCD(cur, dst)$ to find the least common ancestor. If the address of the least common ancestor is smaller than its own address, the current node knows it is in the first segment of routing path and sends packet to its parent. On the contrary, if the address of the least common ancestor is its own address, the current node is in the second segment and sends the packet to the correct child node. The address of the child node can be computed by multiplying cur with the smallest prime factor of the quotient of the address of destination by the address of itself.

```

1  selfGcdValue=seqGCD(cur, dst);
2  for (int i=0; i<number of neighbors; i++){
3      neighborGcdValue[i]=seqGCD(address of neighbor[i], dst);
4  }
5  gNeighborGcdValue = the largest neighborGcdValue;
6  If(gNeighborGcdValue>=selfGcdValue){
7      send packets to the node with gNeighborGcdValue;
8  }else if(gNeighborGcdValue==selfGcdValue){
9      send packets to the parent node;
10 }

```

Figure 4.4: The procedure of setting routing path in the stateful routing protocol.

We here also take Figure 4.1 for instance to demonstrate how node O(12) sends packets to node P(16). Node O(12) first computes $seqGCD(12, 16)$ and is aware that 4 is the address of the common ancestor. Since 4 is smaller than its own address, it sends packets to its parent node F(4). Node F(4) also computes $seqGCD(4, 16)$ and gets the answer as 4, which is equal to its own address. Therefore, the node F computes the smallest prime factor of $(16/4 = 4)$, which is 2, and multiplies this value with its own address 4 to get the address $(2*4=8)$ of the correct child node. When node L(8) receives the packets from node F(4), it also computes $seqGCD(8, 16)$ and sends packets to node P(16) according to the same procedure.

4.2.2 Stateful prime-based self-configured routing protocol

Although stateless routing protocol can configure routing paths without recording any information, the routing paths unfortunately are usually long because nodes find routing paths only from the branches of the address allocation tree. However, there are still some links, like dotted lines in Figure 4.1, not belong to the address allocation tree. If nodes can also choose these links as part of their routing paths, the length of routing path should be shorter. Therefore, we propose stateful routing protocol to shorten the routing path by making each node recording addresses of its neighbors.

In the stateful routing protocol, each node computes the next hop only. When a node with address *cur* receives a packet, it follows the procedures in Figure 4.4 to find the next hop for the destination *dst*. First of all, the node computes the $seqGCD$ for itself and all its neighbors to the destination. If the largest $seqGCD$ value of its neighbors is larger than its own $seqGCD$ value, it sets the neighbor with the largest $seqGCD$ value as the next hop because that neighbor is nearer to the destination. Otherwise, the node sends the packet to its parent node as usual.

We can observe that in Figure 4.1 (d) the routing path from node M to node I is much shorter if we apply stateful routing protocol. For node M(18), its own $seqGCD$ value to node I(15) is $seqGCD(18, 15) = 1$, and all neighbors' $seqGCD$ values are: $seqGCD(16, 15) = 1$ for node P(16), $seqGCD(6, 15) = 1$ for node G(6), $seqGCD(9, 15) = 3$ for node H(9). The largest $seqGCD$ value of neighbors is 3, which is bigger than its own value 1. Therefore, node M(18) sends the packet to the neighbor node H(9).

We need to mention that each node does not need to be aware of it is in the first or second segment of routing path as the procedures in the hop-by-hop method. If the destination is its descendant, it will send the packet to its child naturally since its child is also its neighbor and in most cases that child will have the largest $seqGCD$ value as its own address unless there is a link connect to the destination directly. We also take Figure 4.1 (d) for instance. When node F(4) receives a packet destined to node P(16), it computes $seqGCD(4, 16) = 4$, and $seqGCD$ values of its neighbors are 8, 4, 2 of node L, O, and B respectively. We can notice that node P(16) is a descendant of node F(4), so node L(8), a child of node F(4) and an ancestor of node P(16) has the largest $seqGCD$ value. Therefore, node F(4) sets its child node L(8) as next hop naturally for packets destined to node P(16). Consequently, if there is no neighbor having a larger $seqGCD$ value, the node is in the first segment of the routing path and it can just send packets to its parent directly.

We have proved in Theorem 3 that the proposed prime-based self-configured routing protocols, including stateless and stateful routing protocols, can guarantee loop-free routing paths.

Theorem 3. *Prime-based self-configured routing protocol can guarantee loop-free routing path $N_1, N_2, N_3, \dots, N_k$ from N_1 to N_k .*

Proof. **Case 1: stateless routing protocol:**

In the stateless routing protocol, all routing paths are configuring based on branches of the address allocation tree. Since there is no loop in the branches of a tree, there is no cyclic routing paths.

Case 2: stateful routing protocol:

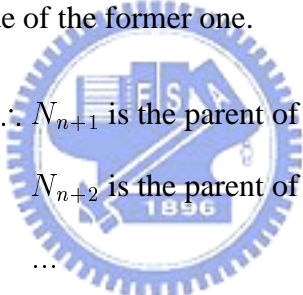
Based on the routing algorithm of stateful prime-based self-configured routing protocol, each node chooses the neighbor with the largest $seqGCD$ of destination as the next hop or sends

packets to its parent if no neighbor has larger seqGCD than its own seqGCD of destination. Therefore, $\forall i < j, seqGCD(N_i, N_k) \leq seqGCD(N_j, N_k)$.

Assume there exists a loop in the routing path $N_1, N_2, N_3, \dots, N_k$, which means $\exists N_m = N_n$ where $m > n$.

$$\begin{aligned} \therefore \begin{cases} seqGCD(N_n, N_k) & = seqGCD(N_m, N_k) \\ seqGCD(N_i, N_k) & \leq seqGCD(N_j, N_k) \forall j < j \end{cases} \\ \therefore seqGCD(N_n, N_k) = seqGCD(N_{n+1}, N_k) = \dots = seqGCD(N_m, N_k) \end{aligned}$$

According to the routing algorithm, a node sends packets to its parent node only if there is no other neighbors having a larger seqGCD to the destination. If two adjacent nodes, N_n and N_{n+1} for instance, in the routing path have the same seqGCD value to the destination, there is no neighbor of the former one (N_n) with a larger seqGCD. In such case, the latter one of two adjacent nodes is the parent node of the former one.



$$\begin{aligned} \therefore N_{n+1} & \text{ is the parent of } N_n \\ N_{n+2} & \text{ is the parent of } N_{n+1} \\ \dots & \\ N_m & \text{ is the parent of } N_{m-1} \end{aligned}$$

Since N_m is an ancestor of N_n , $N_m \neq N_n$. Therefore, there is no loop in the routing path $N_1, N_2, N_3, \dots, N_k$ from N_1 to N_k . □

4.2.3 Performance evaluation

In this section, we study the performance of proposed addressing and routing protocols. We compare the signal overhead and average length of routing paths of our protocol to those of the OSPF (Open Shortest Path First) [11] protocol. OSPF is a link-state routing protocol, in which each host broadcasts its link status to all other hosts. After gathering information from all hosts, each host runs a shortest path algorithm, Dijkstra algorithm [58] for example, to find the shortest paths to all other hosts.

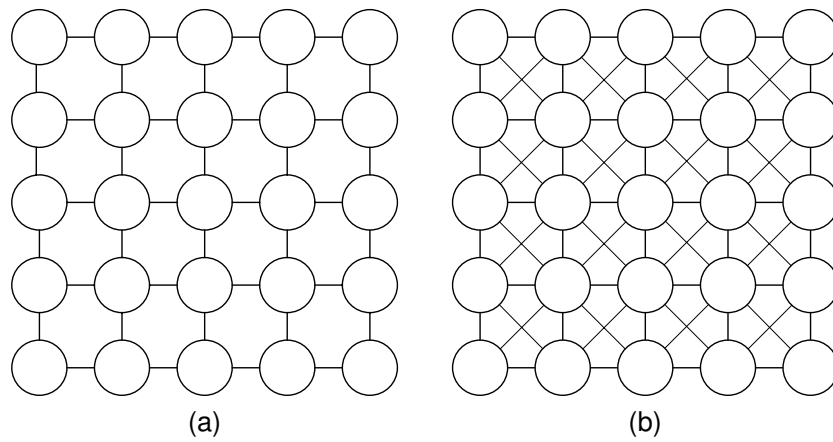


Figure 4.5: Two network topologies when network size=5*5: (a) Link=4 and (b) Link=8.

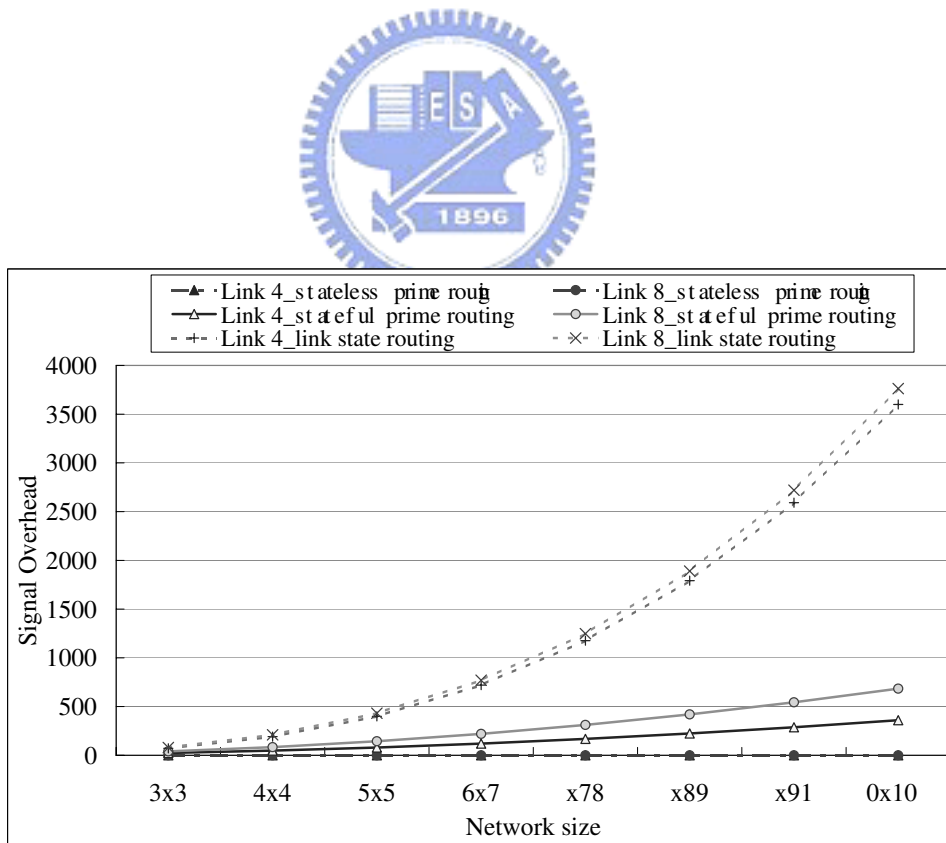


Figure 4.6: The signal overhead.

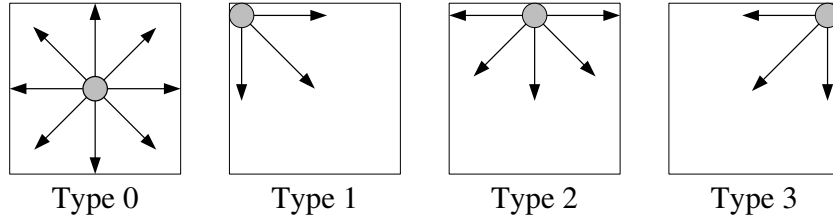


Figure 4.7: Four address allocation types: type0: from central node, type1: from left-top corner node, type2: from center-top node, and type3: from right-top corner node.

We first study signal overhead introduced by our protocol and OSPF under two network topologies as shown in Figure 4.5. In the first topology, each node has links to neighbors in four directions, where each node has links to neighbors in eight directions in the second topology. We use $Link=4$ and $Link=8$ to indicate two different network topologies.

Figure 4.6 presents the signal overhead of two routing protocols in two network topologies. In our proposed prime-based routing protocol, stateless routing protocol does not need to exchange any information and thus introduce no signal overhead; stateful routing protocol needs only detect neighbors' addresses and thus it causes $(average\ degree \times number\ of\ nodes)$ signal overhead. In OSPF, each node needs not only detecting link status but also broadcasting link information to other nodes, so the average signal overhead it introduces is $(average\ degree \times number\ of\ nodes) + (average\ degree \times network\ radius \times number\ of\ nodes)$. We can observe that the proposed routing protocol can significantly decrease signal overhead since it does not need to send routing or link information to other hosts.

Since the proposed routing protocol does not need to exchange information between all nodes in a network, each node does not know the topology of whole network. This might increase the length of routing path, so we also studied average routing length. The proposed prime-based routing protocol is based on the address allocation results of PNAA, different address allocation results may influence the average length of routing paths. We evaluated the average routing length under four different address allocation types by setting different locations of the root node as shown in Figure 4.7. After determining the root node, we assign addresses in counterclockwise direction from its bottom neighbor node. Figure 3.6 is the result of type 1 address allocation method for a 3x3 network, and Figure 4.1 is the result of type 0 address allocation method for a 4x4 network.

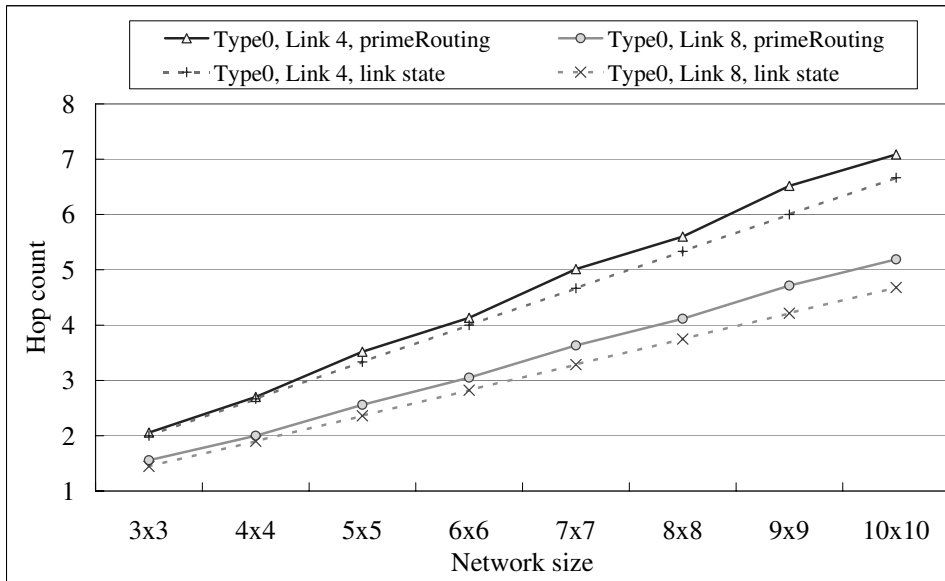


Figure 4.8: The average hop count.

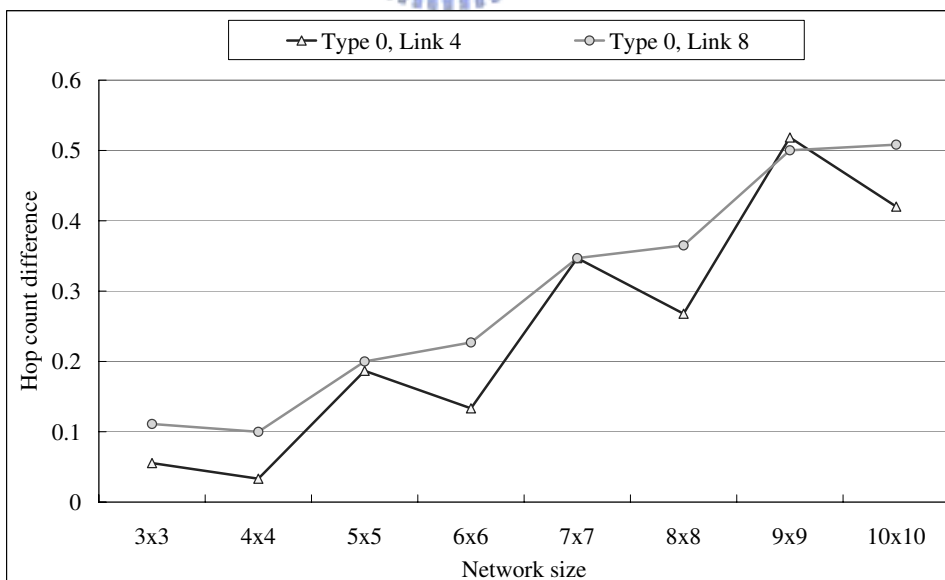


Figure 4.9: The hop count difference when link=4 and type=0.

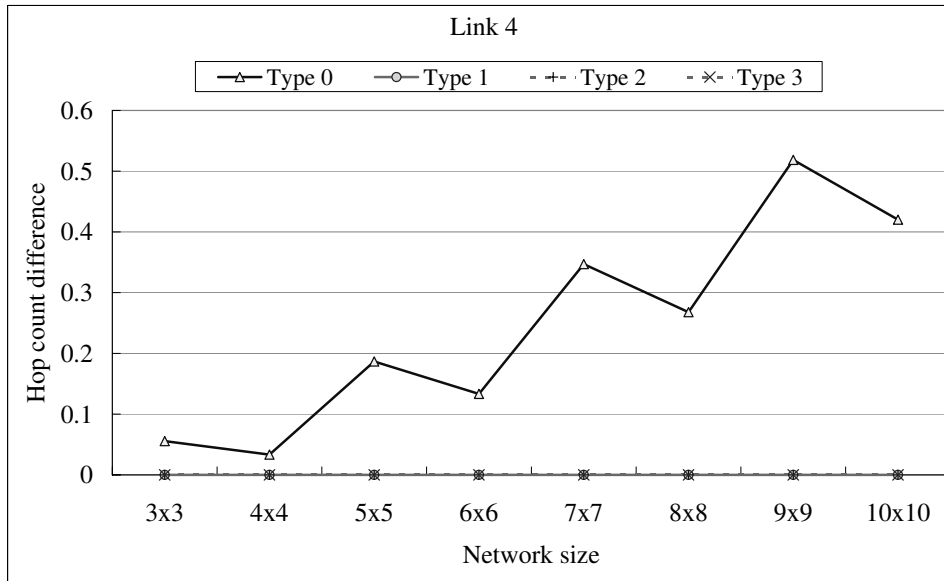


Figure 4.10: The hop count difference of different address assignment type when link=4.

Figure 4.8 illustrates the average path length of both prime-based and link-state routing protocols based on type 0 address allocation result. We observe that the average routing length increases as the network size increases, and the routing length of the first network topology (Link=4) is longer than that of the second topology (Link=8). The routing length of prime-based routing protocol is a little longer than that of link-state routing protocol, but we can find in Figure 4.9 that the difference is always less than one hop.

Figure 4.10 and 4.11 present how different address allocation types influence the average routing length. In a link 4 network topology, only type 0 address allocation method slightly increases average routing length. Type 0 address allocation method generates a star shaped tree, which makes node have a higher probability of finding a farther path by sending packets to the central of network instead of to the corner node directly. Other address allocation types, on the other hand, generate more regular tree, so each node can directly find the shortest path. In a link 8 network topology, there are more links and the topology is more complicated. We can observe from Figure 4.11 that nodes can find shorter routing paths if we assign addresses from the middle of network, like type 0 and type 2 address allocation methods, since these can generate flatter trees.

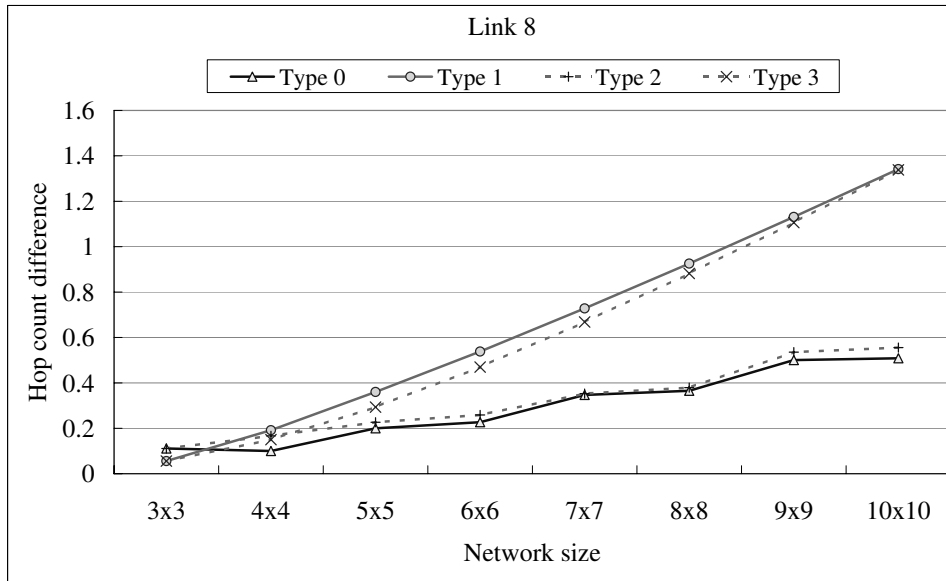


Figure 4.11: The hop count difference of different address assignment type when link=8.

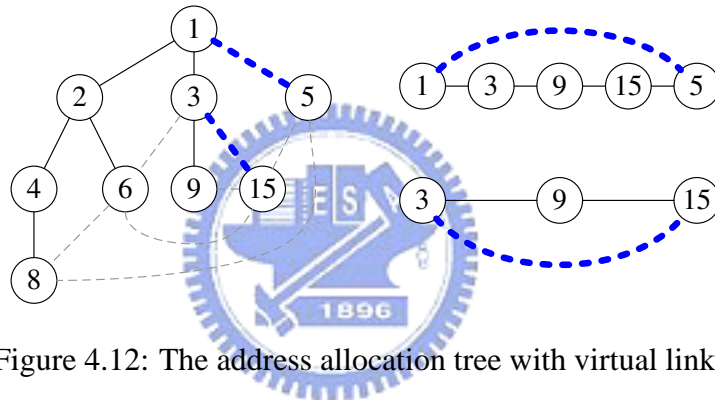


Figure 4.12: The address allocation tree with virtual links.

4.2.4 Discussion

We discuss our prime-based routing protocols in previous sections assuming address space is unlimited. However, some networks might have limited address spaces, so we adjust the proposed protocols to networks with limited addresses in this section.

The adjusted addressing allocation method described in Section 3.3.3 unfortunately violates the address allocation tree structure. The parent node of node H(15) in Figure 3.8 is node C(3), but they do not have a direct link. This causes problems when node H(15) selects its parent node as next hop in original proposed prime-based routing protocols because its parent node C(3) is not its neighbor. Therefore, we add virtual links as shown in Figure 4.12 to solve this problem.

Virtual links can be added during the address confirmation step, which is when the address assigning node sends *DHCP_Ack* message to the new coming node. All nodes along

Table 4.1: Virtual links recorded by nodes in Figure 4.12

Node A (1)		Node I (5)		Node C (3)		Node F (9)		Node H (15)	
Virtual Physical		Virtual Physical		Virtual Physical		Virtual Physical		Virtual Physical	
5	3	1	15	15	9*	3	3*	3	9*
1	1	15	15*	1	1	15	15*	1	9
5	9	1	3	5	5	1	3	5	5
		5	15						

* records for virtual link 3↔15

the address assignment path then add virtual links for both two ending nodes in their *virtual link tables*. For example, when node A(1) sends *DHCP_Ack* to node I(15) along the path (I, H, F, C, A), these five nodes add virtual links for both node I and node A. Table 4.1 shows virtual link tables of nodes all records of nodes for two virtual links 3↔15 and 1↔5. In the following, we will individually detail the adjustments of three proposed routing protocols.

A. Adjusted stateless routing protocol in source routing method

Source node finds the routing path as usual, but checks if there exists a record for the next hop in its virtual link table. If the record is found, it inserts the physical node to the beginning of the routing path. All other nodes receiving packets will do the same checking and insert node to the routing path if necessary. For example, node A(1) sets the routing path to node I(5) as (1, 5), and it finds the physical link to address 5 is address 3. It inserts 3 in the beginning of its routing path and sends the packet to address 3. Upon receiving packets, node C(3) also searches its virtual link table and sends the packet to address 9. Node F(9) sends the packet to node H(15) and then to node I(5) in the same procedure.

B. Adjusted stateless routing protocol in hop-by-hop routing method

We add one step before and one step after the original procedures of stateless routing protocol in hop-by-hop routing method. Before a node finds the next hop, it first checks if there is already a record in its virtual link table for the destination. If there exists a

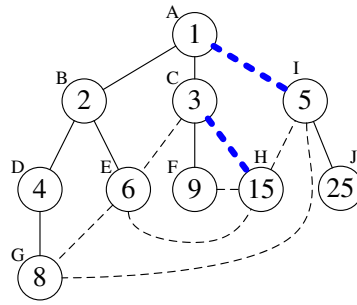


Figure 4.13: Example of address allocation tree with virtual links.

record, the node can send the packet to the recorded physical neighbor directly, like the case that node A(1) finds the next hop for node I(5). Otherwise, the node runs original algorithm to find the next hop, and then check if there is a record for such candidate net hop. If a record matches, the node sends the packet to the physical neighbor instead of virtual neighbor as an instant case of node H(5) finding the next hop for node G(8).

Besides adding prior and posterior examinations, we need to make a minor modification of original routing algorithm. Since nodes record virtual links for only two ending nodes, it may cause problems if an ending node is an internal node of a routing path. For example, if a new node J attaches to node I(5) in Figure 4.12 and obtains address 25 as shown in Figure 4.13, it will cause an infinite loop for node A(1) to find the routing path to node J(25). Node A(1) chooses its child node I(5) as the next hop for destination node J(25), and sends the packet to its physical neighbor node C(3). When node C(3) receives the packet, it has record for node I(5) but no record for node J(25) and sends the packet to its parent node A(1) again after running the routing algorithm, and it falls into an infinite loop. This problem happens when a node select a non-neighboring node as the next hop and such node is not the destination node. In order to solve this problem, we make nodes tunnel packets to its non-neighboring child node, which is not the destination node yet. Node A(1), in the previous example, will tunnel packets to node I(5), and nodes C(3), I(9) and H(15) will search records for node I(5) instead of node J(25). After all, the routing path will be correct as (A(1), C(3), I(9), H(15), I(5), J(25)).

C. Adjusted stateful routing protocol

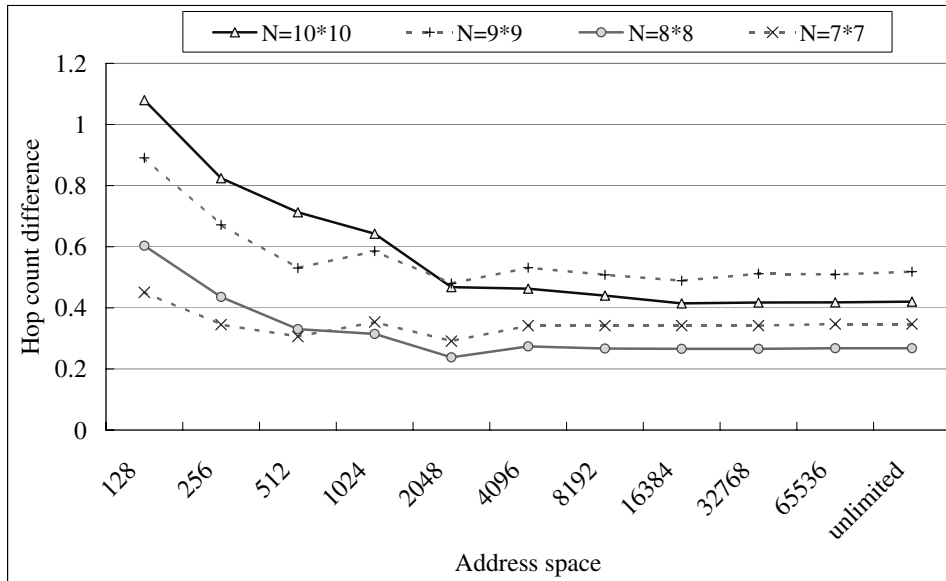


Figure 4.14: The hop count difference with limited address space.

The adjustment of stateful routing protocol is the same as that of stateless routing protocol in hop-by-hop method. We add two examinations before and after the original stateful routing algorithm, and we also make nodes tunnel packet to their non-neighboring internal nodes.

In order to make our protocols fit address-limited networks, we adjust proposed addressing and routing protocols, but this might sacrifice the routing length. Figure 4.14 is the effect of address space on the routing length for stateful routing protocol. If address space is bounded in a smaller range, the difference in routing length between our method and Dijkstra algorithm is larger. Fortunately, the difference is still less than one hop and reaches the ideal value, which is the difference value without address limitation, in some limited address space. For example, a 7x7 network needs only 512 addresses to have the ideal routing length.

Limited address spaces may also influence address allocation latency. If a DHCP proxy needs to relay requests to its parent node, the response time will be longer and the new coming node should extend its timer to wait for a longer period. Therefore, the value of timer at a new coming node determines how far the node can ask for an address. We set the value of timer as multiples of average round trip time between two adjacent nodes, and denote the multiple value as *level*. If level is equal to one, the new coming node can only have an address offered by its neighbors. A smaller level will limit the flexibility of address allocation and may need

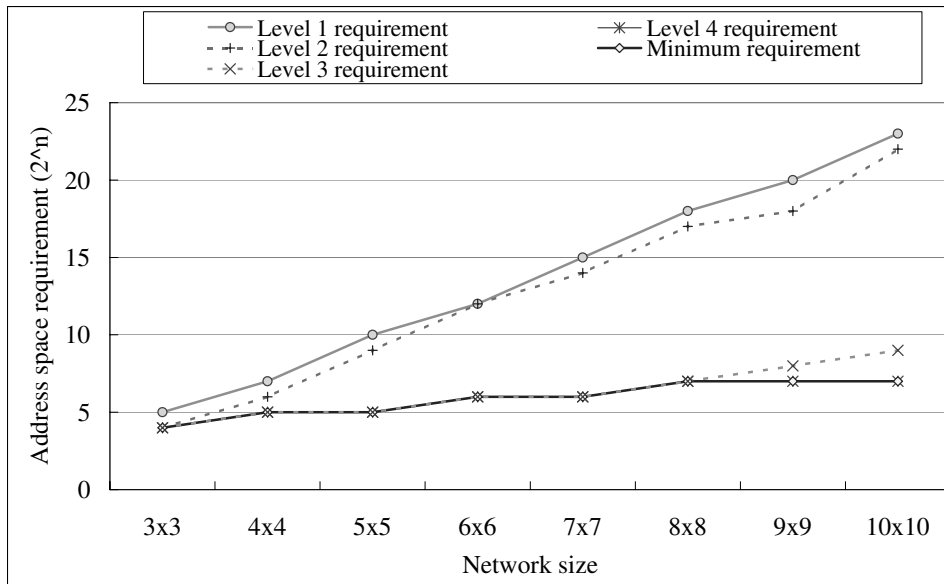


Figure 4.15: The effect of levels for help on the address space requirement.

a larger address space. Figure 4.15 gives us an idea that how many addresses are required with different levels. A network with fixed size has the minimum address requirement, e.g., a 3x3 network needs at least 2^4 addresses. Observing Figure 4.15, we only need the minimum address requirement when we set level larger than 3.

In a summary, adjusted addressing and routing protocols can solve the problem of limited addresses while sacrificing minor advantage of routing length and address allocation latency. From the simulation results, we can find that adjusted protocols can still perform as well as original protocols if providing address space larger than 2^{10} for a network with size smaller than 10x10. Besides, the latency can also be bounded in 3 times of round trip time between two adjacent nodes.

4.3 Summary

In this chapter, we described a prime-based self-configured routing protocol to solve internal routing problems in multi-hop networks with fixed topologies. The proposed routing protocol makes use of logical relationship of addressing results of proposed PNAA in Chapter 3 to enable mobile hosts configure routing paths by themselves. The proposed prime-based self-configured routing protocol can significantly decrease both signal overhead caused by maintaining routing information periodically and path setup time by soliciting routing paths.

Chapter 5

External Routing Mechanisms for Hierarchical Mobile Networks

In the previous chapter, we solved internal routing problems for mobile nodes. Besides internal routing, we also need to solve external routing problem, which means we need to let each mobile host be able to route packets to a corresponding node in the Internet no matter which network the mobile host stays in. In 5.1, we describe the overall architecture of external routing. We can observe that gateways are always the bottlenecks of traffic between different networks. In Section 5.2, we proposed a load-balanced routing protocol to balance loads between gateways of a network. To provide host mobility, we apply and modifies Mobile IP to make it fit our architecture in 5.3.

In addition, location may also help to enhance routing performance. For example, when a mobile host roams into a new network, it needs to perform registration process before being able to transmitting packets. If the mobile host has location information of itself and other access routers, it can perform pre-registration to reduce handoff delay. In Section 5.4, we will propose a localization algorithm and describe how location information can enhance routing performance.

5.1 External Routing Architecture

At least one gateway is configured in each network to help internal nodes communicate with corresponding nodes outside the network. There are three MANETs and three gateways in Figure 5.1. The first MANET communicates with other nodes via gateway A. Nodes in the second MANET can connect to the Internet via either gateway B or C. The third MANET is a

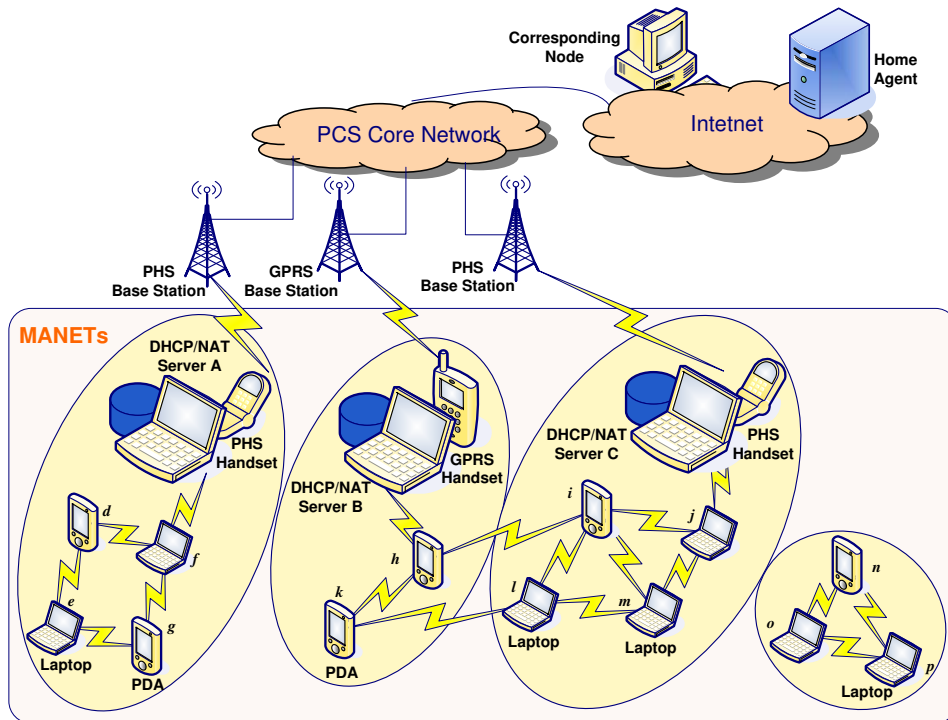


Figure 5.1: External routing architecture.

stand alone one since it does not have a gateway.

5.2 Load-balanced routing protocol

A MANET is typically considered as a stand-alone network, but it is important to enable its Internet accessibility. On one hand, users in a MANET can enjoy the tremendous resources in the Internet. On the other hand, the connectivity between multiple MANETs may be greatly improved. For such connectivity, several works [2, 63, 2, 20, 39] have proposed possible architectures by deploying gateways to help mobile hosts route packets to the Internet. Among these approaches, some takes a proactive approach by modifying DSDV [63], some takes a reactive approach by modifying AODV [20, 2], while some takes a hybrid approach [39].

The system architecture of how to extend connectivity of MANETs to the Internet is shown in Figure 5.1. We can observe from the figure that gateways are always bottlenecks of external traffic. We propose a load-balanced routing protocol in this section to balance traffic between all gateways.

5.2.1 Protocol design

A load-balancing routing protocol is designed to help mobile hosts choose their gateways. We propose Minimum Load-Index (MLI) routing protocol [23, 3]. MLI tries to dynamically establish boundaries between gateways' service ranges by taking the load indices of gateways and the traffic loads of hosts into account. This scheme is fully distributed and can be run by each host independently. Each gateway g will periodically broadcast advertisement messages containing its current load index (L_g), which is the ratio of the traffic load of its high-tier interface to the maximum bandwidth of its high-tier interface. Each host x should keep a record of the load index of its current serving gateway. When a host x hears an advertisement from any gateway g , the following rules are executed:

1. If x currently has no serving gateway, it chooses g as its serving gateway by recording g 's current load index and setting the host leading to g with the shortest distance as its default gateway. Then x rebroadcasts the advertisement.
2. If g is already x 's serving gateway, x should update g 's index as necessary and rebroadcast the advertisement.
3. If g is different from x 's current serving gateway, say g' , then x will update g as its serving gateway with a predefined gateway-switching probability P_g only if x has accepted g' as its serving gateway for over a time period τ and $L_g + \frac{T_x}{C_g} + \Delta_l \leq L_{g'} - \frac{T_x}{C_{g'}}$, where T_x is the traffic index of x , C_g is the total capacity of the high-tier interface of g , and Δ_l is a predefined gateway-switching threshold.

The above steps are similar to a diffusion process. A gateway with less traffic load will extend its service range, while one with more load will shrink its service range. Note that in the above rule 3, we do not require x to rebroadcast the advertisement, no matter x accepts g as its new gateway or not. We do this on purpose so as to achieve a slow diffusion. Otherwise, a gateway with a very low load will quickly take too many hosts and lead to a network fluctuating situation. As a result, the service ranges of gateways will be extended at most one hop in each advertisement. P_g and Δ_l are designed with the similar purpose.

Taking Figure 5.2 for example, there are three sub-MANETs with load indices 0.5, 0.8, and 0.6, respectively. Then n will choose C as its serving gateway according to rule 1. Suppose

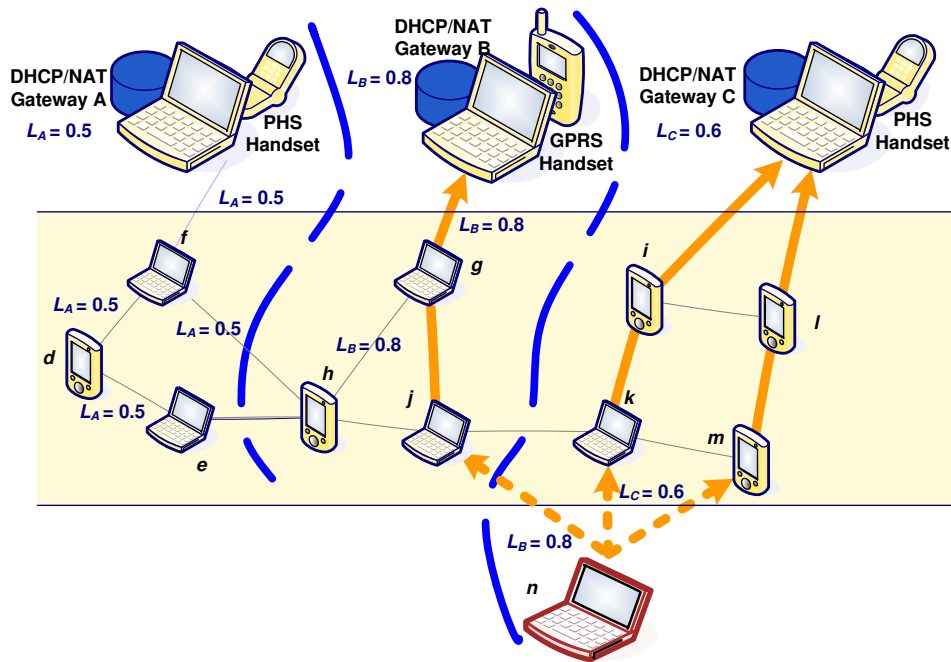


Figure 5.2: An example of load-balanced routing protocol.

that n has a very high traffic load and thus saturates sub-MANET C . Then this may force host k to move to sub-MANET B .

5.2.2 Performance evaluation

We have implemented a prototype in an environment with five IBM laptops, and each with an 802.11b NIC working in ad hoc mode. Two of them equipped two PHS handsets are gateways. The operating system is the Windows 2000 Advanced Server supporting the following services. First, we activate the “Internet Connection Sharing” service on gateways to share their Internet connections with other hosts. Second, the “DHCP Server” service is also opened in each gateway. Third, the “Routing and Remote Access” service is activated in each host to offer routing services and to drive the TCP/IP forwarding engine.

We embed our MLI routing scheme in DSDV [43]. In DSDV, hosts exchange routing tables with neighbors. We create a MLI routing table with two more fields: *gateway indication* and *loading*. The former is to notify whether this routing entry is generated by a gateway or not, and the latter is to record the load index and capacity of the gateway. We wrote a MLI daemon for hosts to periodically exchange and update their MLI routing tables. Besides, the MLI daemon is also responsible for updating the system routing table that is actually used

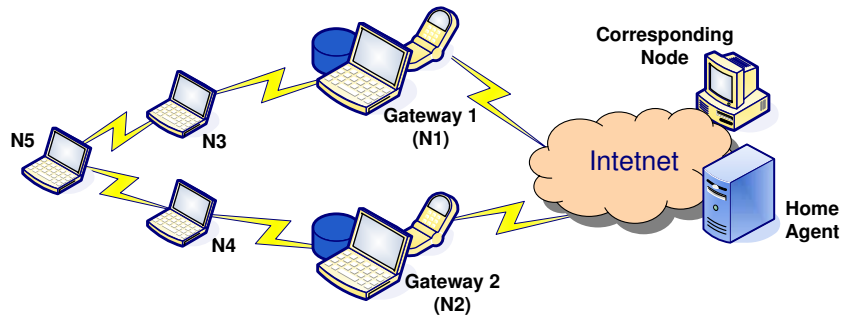


Figure 5.3: The testing environment.

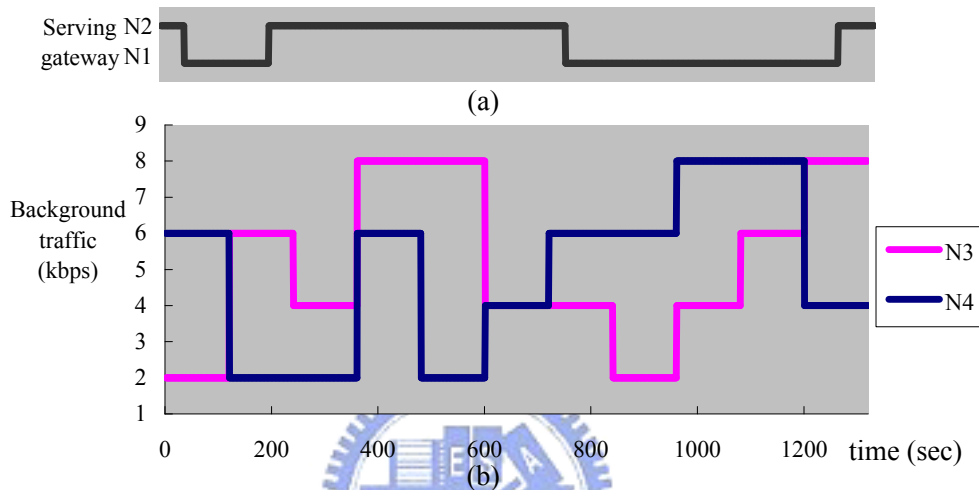


Figure 5.4: (a) selection of gateway by host N5, and (b) traffic loads at gateway N1 and N2.

to forward packets in each host. Whenever the MLI daemon decides to change gateways, it updates the routing entry of its home agent in the system routing table.

We present some performance results observed from our implementation testbed. The testing environment is shown in Figure 5.3. In the beginning, there are two separate MANETs: N3 connected to gateway 1 (N1) and N4 connected to gateway 2 (N2). Then a new host N5 arrives, which can connect to both N3 and N4 and needs to communicate with a corresponding node (CN) in the Internet. We run a FTP session between N5 and CN. We also generate some random background traffic ranging from 2 Kbps to 8 Kbps from N3 and N4 to CN, and study the throughput between N5 and CN.

Figure 5.4 (a) illustrates how N5 chooses its serving gateway, given the traffic loads of gateways in Figure 5.4 (b). We observe that in general N5 can choose the lightest-load gateway without problem. However, there may be some delay before the right gateway can be selected because of the broadcast periods of gateways (here we set the period to be 10 seconds).

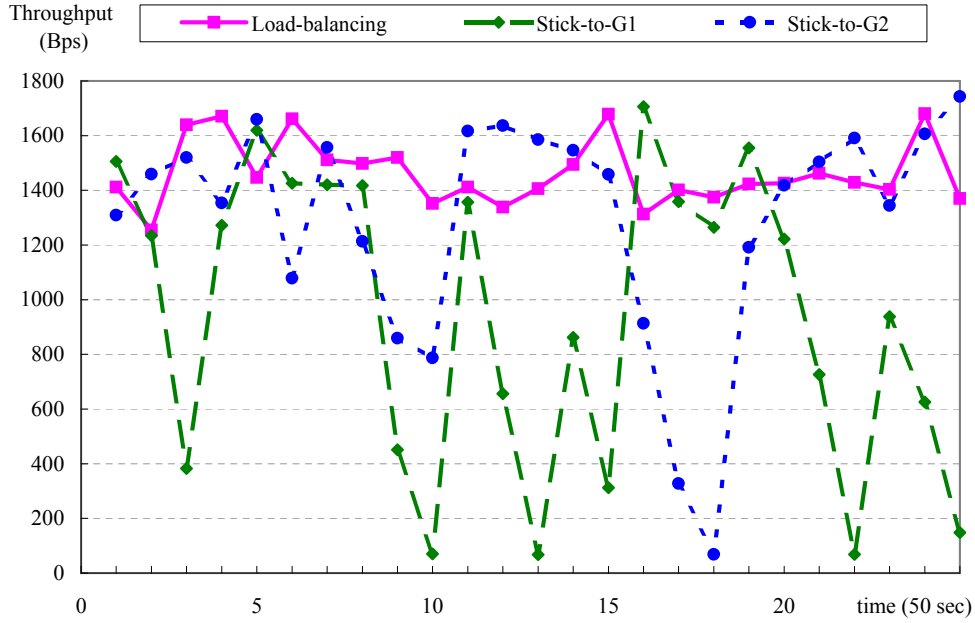


Figure 5.5: Trace of N5's throughput over time.

Table 5.1: The impact of traffic fluctuation to load-balancing protocol.

Variation period	60 sec.	90 sec.	120 sec.	150 sec.
Load-balancing	1176.42	1342.65	1446.23	1419.69
Stick-to-N1	1357.91	1445.48	1121.95	1377.74
Stick-to-N2	1400.30	1403.27	1286.68	1246.59

Figure 5.5 traces the throughput of N5 when our load-balancing routing protocol is applied or not. The traffic being generated is the same as Figure 5.4 (b). We see not only seamless handoff of N5 between gateways, but also quite stable throughput connecting to CN. On the contrary, the throughput of N5 fluctuates if N5 sticks to N1 or N2.

We mentioned earlier that the broadcast period of gateways may delay a host's gateway selection. Here we set the broadcast period to 10 seconds and $\tau = 0$ second, but vary the traffic loads of N3 and N4 every 60, 90, 120, and 150 seconds. As shown in Table 5.1, when the background traffics fluctuate too fast (variation period below 90 seconds), N5 may not choose the correct gateway in time. The degradation of average throughput is due to handoff overheads (such as Mobile IP re-registration, during which no packet will be sent to CN, thus causing waste of bandwidth).

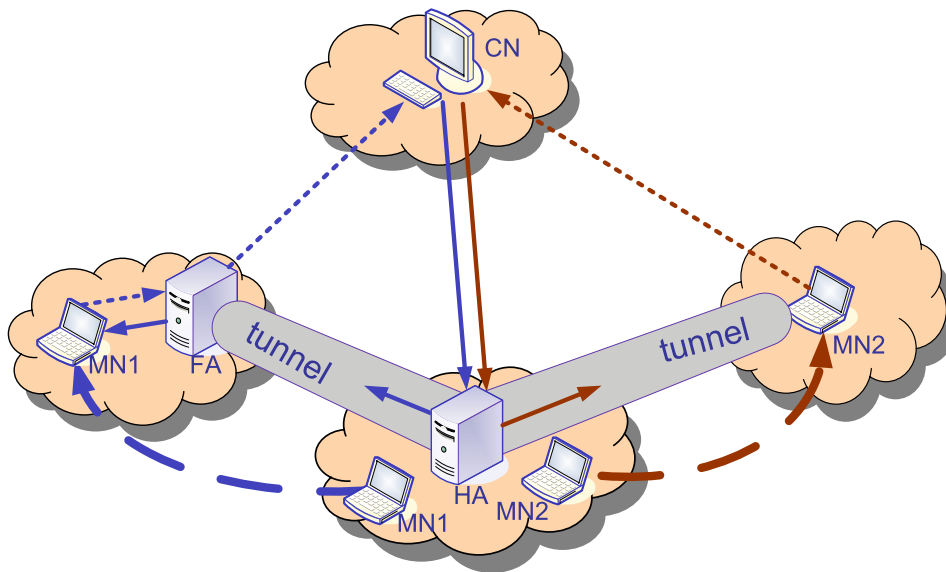


Figure 5.6: Mobile IP operation scenarios.

5.3 Mobile IP Adaption

Mobile IP [6] is a good solution to supporting seamless connection for mobile hosts while roaming. Mobile IP works by allowing a mobile node (MN) to be associated with two IP addresses: a *home address* and a dynamic *care-of address*. The home address is fixed, and the care-of address changes at each new point of attachment to the Internet. The home IP address assigned to the mobile client makes it logically appear as if the mobile node is attached to its home network. The care-of address identifies MN's current topological location. For a correspondent node (CN), the MN seems to be attached to its home network all the time.

As shown in Figure 5.6, two mobility agents are deployed by mobile IP. Home Agent (HA) locates in a home network to receive traffic destined to MN's home IP address when MN is not physically attached to the home network. When MN attaches to a foreign network, HA tunnels traffic to a Foreign Agent (FA) using MN's current care-of address like MN1 in Figure 5.6. Whenever MN moves its point of attachment, it registers a new care-of address with its Home Agent.

HAs and FAs regularly broadcast agent advertisements. When an MN receives an agent advertisement, it can obtain an IP address. The MN may also broadcast an agent solicitation that will be answered by any FA that receives it. Thus, agent advertisement procedure allows for the detection of mobility agents, lets the MN determine the network number and status of

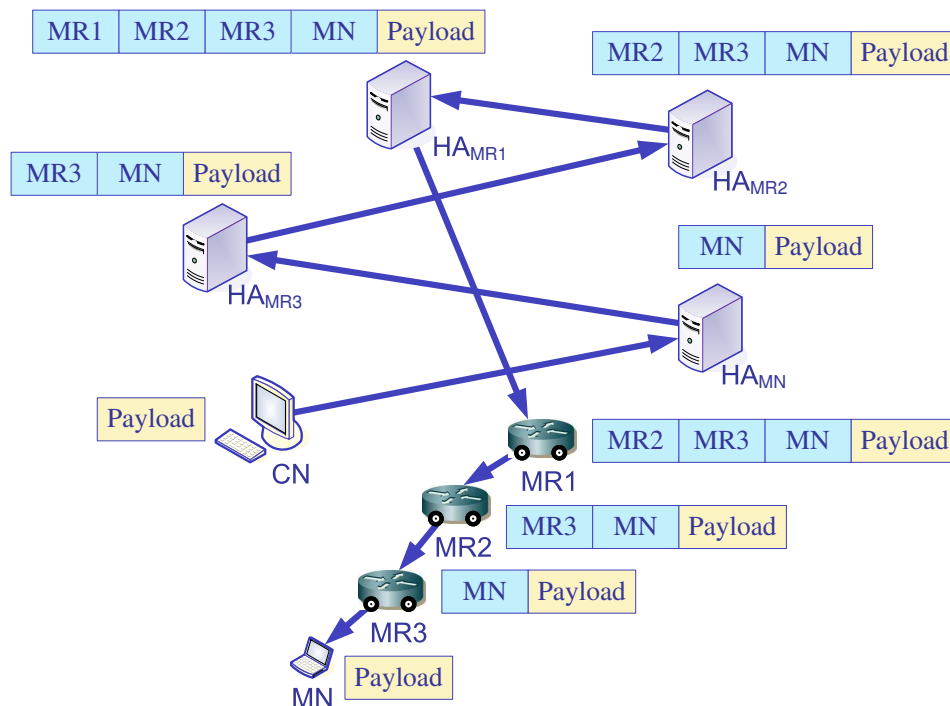


Figure 5.7: An example of ping-pong routing for an MN attaches to a hierarchical mobile network.

its link to the Internet, and identifies whether the agent is a HA or a FA. Once an MN receives a care-of address, a registration process is used to inform the HA of the care-of address. The registration allows the HA to update its routing table to include the mobile's home address, current care-of address, and a registration lifetime.

Mobile IP provides another alternative to let MN have a co-located care-of address without support of FAs. In this case, MN like MN2 in Figure 5.6 needs to detect movement by sending router solicitation periodically. Once it finds it moves into a new network, it needs to acquire an co-located care-of address, which can be obtained by running DHCP client. The registration process will also be triggered after MN having an address. Afterwards, HA tunnels traffic destined to MN to its co-located care-of address directly.

We can apply Mobile IP to solve roaming problems when an MN roams into a MANET. However, it may cause polygon or ping-pong routing when MNs roam into a hierarchical mobile network. As shown in Figure 5.7, an MN attaches to MR3, which is in a hierarchial NEMO with ancestor mobile routers MR2 and MR1. When CN sends a packet to MN, the packet will be routed to MN's home network and be intercepted by HA_{MN} . HA_{MN} checks its routing table and tunnels the packet to MN's care-of address, which is MR3's home address.

The packet then will be tunneled to $HA_M R3$, $HA_M R2$, $HA_M R1$ in order. At last, the packet will be tunneled to MR1. MR1 decapsulates the packets and forward to MR2, and then to MR3 and MN similarly. This problem can be solved by Reverse Routing Header [40], which records the path from MN to CN. Therefore, CN can sends packets to MN by proceeding the reverse routing path from MN to CN.

5.3.1 Problems of roaming in public networks

There is a problem for LFNs in a NEMO. LFNs not like MNs have only one address. While CN sends a packet destined to an LFN, the packet will be routed to MR's home network. Unfortunately, LFN has left there already, and the HA of MR does not have any record for the LFN, either. In the specification of Mobile IP, LFN could be treated as an MN, so HA will have records of all LFN in the NEMO. However, each LFN in the NEMO needs to send registration request as a normal MN does whenever the NEMO roams into a new network. This may cause huge overhead and HA also needs to store lots same record. Thus we propose a method to reduce signal overhead.

The proposed method makes MR to register for all LFNs in its subnetwork by sending its network prefix to it HA only. As a result, HA needs to store only one more record for each MR. Whenever the HA receives a packet destined to any node belongs to a subnetwork with the same prefix it records in its routing table, it just tunnels the packet to the care-of address of MR as recorded in its routing table.

5.3.2 Problems of roaming in private networks

Although Mobile IP [6] can maintain seamless connection for MNs while roaming, mobile nodes are limited to roam in public networks. One reason is that HAs can not tunnel packets to a private IP address. Another reason is that the original IP-in-IP tunnels hide port information, making Network Address Translation (NAT) servers unable to do the translation. However, it is normal to configure a MANET as a private network like Figure 5.1, and it will cause problem if an MN roams into a private MANET. To maintain correct tunnels to mobile hosts in private networks, we adopt a mechanism called *NAT traversal* [19]. It applies UDP tunneling, which encapsulates an extra UDP/IP header outside the IP payload, to provide port information to

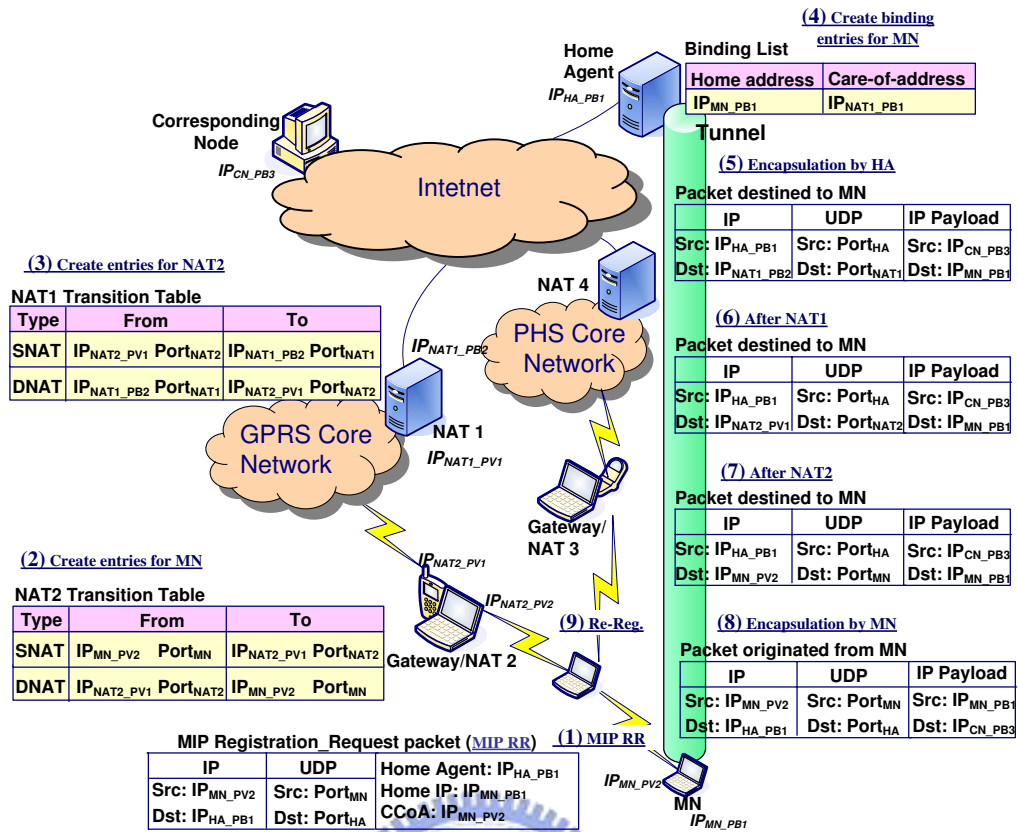


Figure 5.8: Operations of NAT traversal.

NAT servers.

Figure 5.8 shows how NAT traversal works. There are four NAT servers and two gateways (one using GPRS and one using PHS). $NAT1$ is the NAT server in the GPRS core network, to which $NAT2$ is connected, and $NAT4$ is the server in the PHS network, to which $NAT3$ is connected. IP_{A_PB}/IP_{A_PV} denotes the IP address of a host A when it is in a public/private domain. For example, the GPRS gateway has two interfaces: IP_{NAT2_PV2} on the low-tier network and IP_{NAT2_PV1} on the high-tier GPRS network.

Consider a mobile node MN , which has a permanent address IP_{MN_PB1} and a home agent (HA) with an address IP_{HA_PB1} , in a public network $PB1$. Suppose that MN moves into a MANET and obtains a CCoA IP_{MN_PV2} assigned by the gateway $NAT2$. MN will send a *registration_request* to inform HA its new care-of address via its serving gateway (step 1). When forwarding the request, $NAT1$ and $NAT2$ will update their transition tables (steps 2 and 3). When HA receives this *registration_request*, it compares its source address against the CCoA that it claims. If they are not compatible, HA considers MN as inside a private network

and registers IP_{NAT1_PB} as MN 's care-of address (step 4). Later on, for each packet from CN destined to MN , an extra IP/UDP header will be added (step 5). Both $NAT1$ and $NAT2$ will translate the packet according to their DNAT transition rules (steps 6 and 7). Similarly, whenever MN wants to send a packet, it also encapsulates an IP/UDP header (step 8). The SNAT transition rules will then be used to forward the packet to HA (we omit the details here). On receipt of the packet, HA then decapsulates the IP/UDP header and forwards the packet to CN .

When MN roams into a different sub-MANET, we allow it to keep its care-of address. However, MN has to re-register with HA (this is not required in typical Mobile IP); otherwise, HA and the NAT servers will not have correct transition rules. For example, when MN moves to $NAT3$, a re-registration will be sent to HA (step 9).

5.4 Location-assisted routing enhancements

Besides balancing traffic between gateways and scheduling traffic to improve routing performance, location information can also help mobile host to route packets correctly or reduce handoff latency while roaming. In this section, we present how to locate mobile hosts or mobile routers and how location information can enhance routing performance.

5.4.1 Localization algorithm

There are lots of localization algorithms [15, 38, 62, 65, 47, 22, 27, 28] to locate mobile hosts. The easiest way is making each mobile host equip a Global Positioning System (GPS) module. However, GPS modules are still expensive and they are limited in outdoor environments. Other localization algorithms usually estimate location according to some nodes with fixed or known positions. In general, there are two steps to locate a mobile host. The first one is to estimate the distance of the mobile host to fixed nodes. As shown in Figure 5.9, distances from MN to $BS1$, $BS2$, and MR need to be estimated in the first step. The distance can be estimated according to received signal strength (RSS), time of arrival (TOA), or round trip time (RTT). After estimating distances, the location of MN could be calculated in the second step. At least three equations are necessary to resolve (x, y, z) , so at least three reference points are required

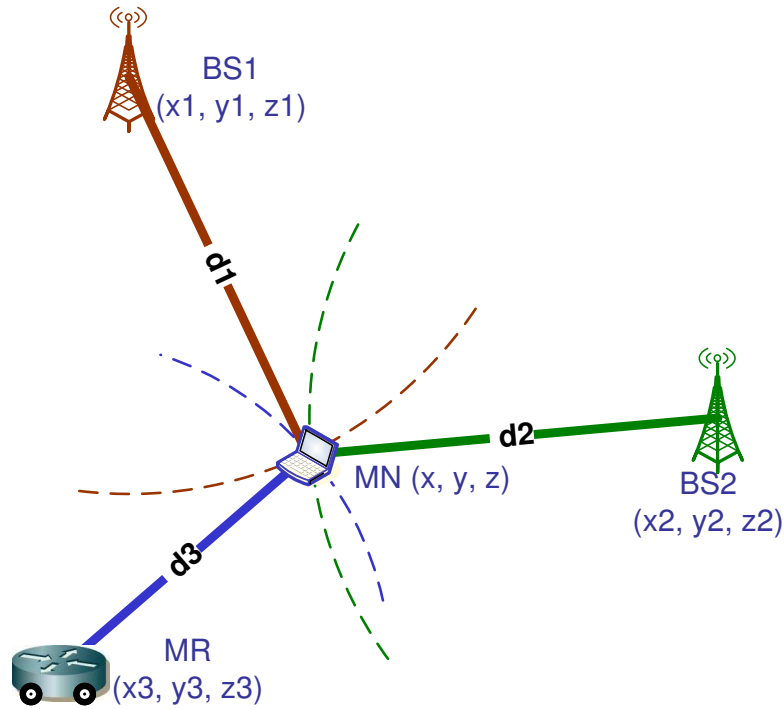


Figure 5.9: Illustration of locating an MN.

to locating an MN.

$$d_i = \sqrt{(x_i - x)^2 + (y_i - y)^2 + (z_i - z)^2} + \varepsilon$$

In the architecture of hierarchical mobile network, we make base stations and MRs with GPS modules as reference points as shown in Figure 5.9. All other MNs or MRs without GPS modules can be located by these reference points. We take RSS for example to describe the scenario of locating the MN. MRs with GPS modules need to report their location to a location server, which has location information of all reference points. The MN periodically detects strength of received signal from all nodes and sends detected information to the location server. After calculation, the location server can recognize where the MN is.

5.4.2 Location-based fast handoff

Once we know the location information of all MRs and MNs, we can make MNs perform pre-registration to proper base stations or MRs. All mobile hosts update their detected signal strength to the location server periodically. When an MN detect that the received signal strength of current attach point is beneath a threshold, it sends request to the location server to

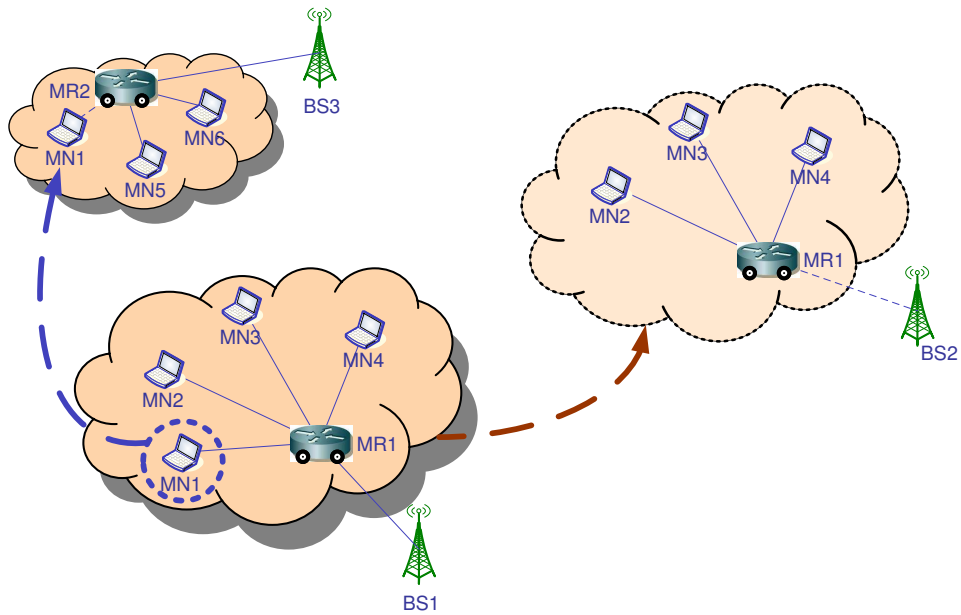


Figure 5.10: An example of selecting candidate points for pre-registration.

ask for candidate points to perform pre-registration.

There are two NEMOs forming by MR1 and MR2 in Figure 5.10. MR1 currently attaches to BS1 and MR2 attaches to BS3. MR1 detects the decrement of signal strength from BS1 while moving and sends a request to the location server. Location server searches location database and reply candidate points, including BS2 and BS3, to MR1. The other moving node MN1 attaching to MR1 also sends a request to the centralized server as it gradually departs the transmission range of MR1, and receives response with candidate points MR2. Note that MNs except MN1 moves with MR1, so they can still receive high quality of signal strength of MR1 and will not trigger pre-registration process. This can reduce unnecessary signal overhead caused by pre-registration.

5.5 Summary

This chapter proposed several mechanisms to solve external routing problems when MNs roams between different networks in a hierarchical mobile network. Generally, a gateway can be configured in each network to help nodes communicate with corresponding nodes outside the network. We proposed a load balanced routing protocol to balance traffic loads among multiple gateways and prevent any one gateway from being the traffic bottleneck. A prototype is implemented on Windows platform to verify our architecture and testing results do show

the benefit of load-balancing routing. We also modify Mobile IP to support host mobility in hierarchical mobile networks. Besides, a localization algorithm is proposed and the location information can help mobile hosts perform pre-registration to reduce handoff delay.



Chapter 6

Information Provisioning Mechanisms for Hierarchical Mobile Networks

After solving basic addressing and routing problems in hierarchical mobile networks, we need to provide services to users. This may cause two problems. The first one is that the external bandwidth of gateways are shared by all users in the same network, and such scarce bandwidth usually is the bottleneck of external traffic. Therefore, we propose a load-based scheduling mechanism to further help gateways efficiently utilize external bandwidth in Section 6.1. The second problem is that in the environment of hierarchical mobile networks, information might be distributed all over the network or in various mobile devices of users while wireless accesses provides high flexibility and convenience to users. However, this may cause problems for users to manage their personal information. We need mechanism to store information and help users manage and access their information efficiently. The proposed mechanism will be described in Section 6.2

6.1 Load-based scheduling mechanism

We mentioned in Chapter 2 that although MONETs can extend our accessibility to the Internet, some problems [60, 56] still remain to be solved before the MONETs can be widely deployed. Among these problems, performance issues need to be considered when lots of users share the limited bandwidth, especially when a MONET moves in a high speed. When the MONET moves fast, a high-tier network like GPRS or WCDMA/cdma2000 in the future must be applied. However, the low data transmission rate of the high-tier network might not be enough to serve all nodes in the MONET. Therefore, we focus on the performance issues

and study how to utilize the limited wireless bandwidth of the MONET more efficiently in this section.

In this section, we will propose a hierarchical-proxy architecture with a load-based scheduling scheme [24] that can utilize scarce wireless bandwidth efficiently to fulfill the requests issued from the nodes in MONETs.

6.1.1 Two-tier architecture

A two-tier wireless network can provide a broad coverage with the high-tier wireless network and a high data transmission rate with the low-tier wireless network. When the MONET moves slowly or stops within the coverage area of a low-tier wireless Access Point (AP), it may access the Internet through the low-tier wireless interface to provide high bandwidth to the local nodes. On the other hand, when the MONET moves fast or is outside of the coverage area of the low-tier network, it may attach to the high-tier wireless network to provide continuous Internet access to the local nodes. The deployment of two-tier wireless networks has been described in [16]. Without loss of generality, we assume that the low-tier networks are built only at the stations. Therefore the MONET can access the Internet only through the high-tier network when the train moves outside the stations. As a consequence, a two-tier MR needs to be equipped with at least three interfaces, two external interfaces and one internal interface. The MR can thus attach to the Internet through the two external interfaces and communicate with local nodes through the internal interface.

Figure 6.1 shows the hierarchical proxy and two-tier architecture. There are three hierarchical proxies, denoted as Local Proxies (LP), Station Proxies (SP), and Global Proxies (GP). An LP is located within a MONET and is responsible for caching the content retrieved from the Internet or buffering the requests that are suspended for some reason (to be explained later). A Global Proxy (GP) is situated at some place in the Internet and can cache contents retrieved from the Internet to shorten the wired-side Internet access delay. If the GP is far away from a station, an SP can be deployed at the station to further shorten the wired-side communication delay. That is, the SP can cache the information forwarded from the GP and rapidly transferring, through the low-tier wireless network, the information to the LP in a MONET when the MONET arrives at the station. Furthermore, because SPs are fixed hosts that can access the

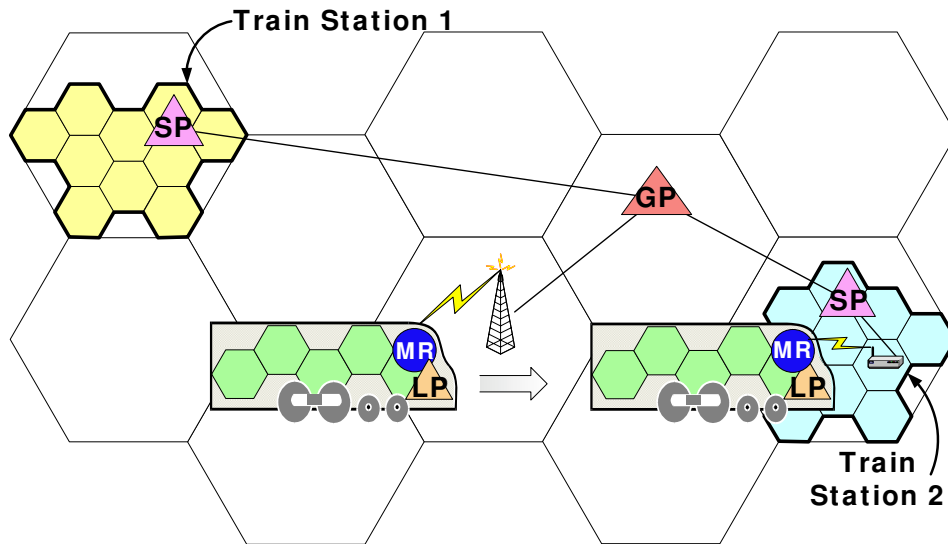


Figure 6.1: The hierarchical-proxy architecture for two-tier wireless networks.

Internet through the wired network, SPs can access information directly from any servers in the Internet. Therefore, the GP may distribute its load by forwarding requests to some appropriate SPs. The SPs can then access information directly from some servers in the Internet and later deliver the information to the destined LPs through the low-tier networks. In the case where the GP can deliver information to stations quickly enough, SPs may not be necessary to the load-based scheduling scheme.

6.1.2 Load-based scheduling scheme

In this section, we describe the detailed operation scenarios of the load-based scheduling scheme in a multi-service environment, which provides a variety of services such as web browsing, FTP, e-mail and etc. In sections 3.1, we present the operation scenarios in the load-based scheduling scheme when SPs are not deployed in the stations. We then revise the operation procedures of the load-based scheduling scheme in section 3.2 for the case where SPs are deployed in the stations so that we can further shorten the wired-side delay or distribute the load of the GP.

We first classify services into lightweight services and heavyweight services according to the data size of responses to the requests of services. For example, web and telnet services would likely be the lightweight services and FTP services the heavyweight services. For the lightweight services, users would expect to receive responses within a small tolerance time be-

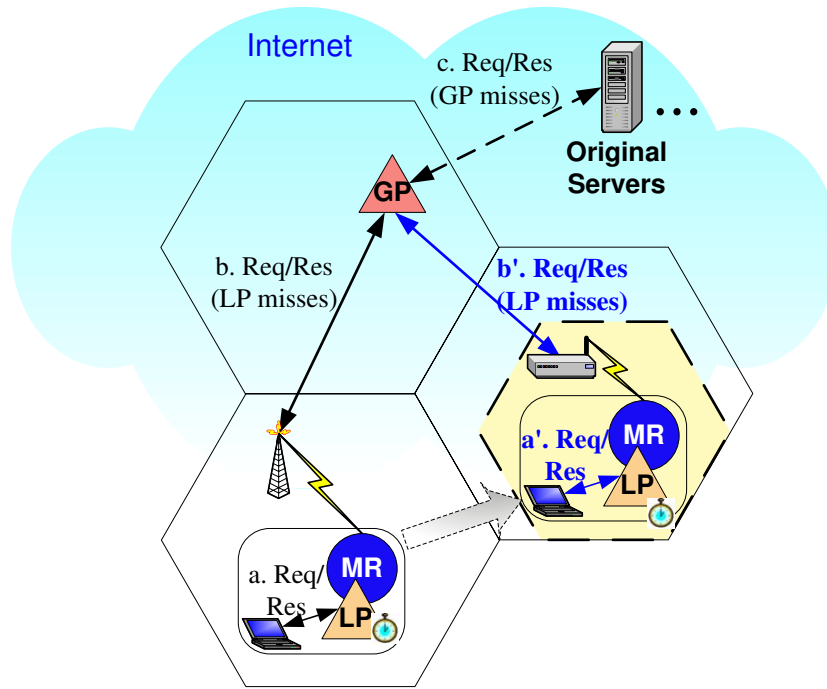


Figure 6.2: The load-based scheduling scheme without SPs.

cause of the intrinsic interactive-nature of the lightweight services. However, for heavyweight services, users could normally tolerate longer time to get the desired responses just in time or before the time they leave the MONET. Therefore the load-based scheduling scheme aims to use the wireless bandwidth more efficiently to transfer both heavyweight and lightweight responses within the respective tolerance time. The underlying idea of the proxy-based scheduling scheme is to reserve scarce high-tier wireless bandwidth to the lightweight services and avoid wasting wireless bandwidth for those requests that can never be transferred to the users within their tolerance time.

1. Load-based scheduling scheme without SP

If SPs are not deployed in stations, the load-based scheduling scheme can make use of the GP and LPs to leverage the performance of MONETs. Each LP is equipped with a timer to guarantee a maximum time delay. The responses could be the results or the status of the requests according to the types of responses. Figure 6.2 represents the network topology without SPs. When a MONET moves between stations, the MR attaches to the high-tier network. Users behind local nodes send requests to the LP and receive results if cache hits in the LP (process a). Otherwise, the LP forwards requests to the GP. If the GP has the requested

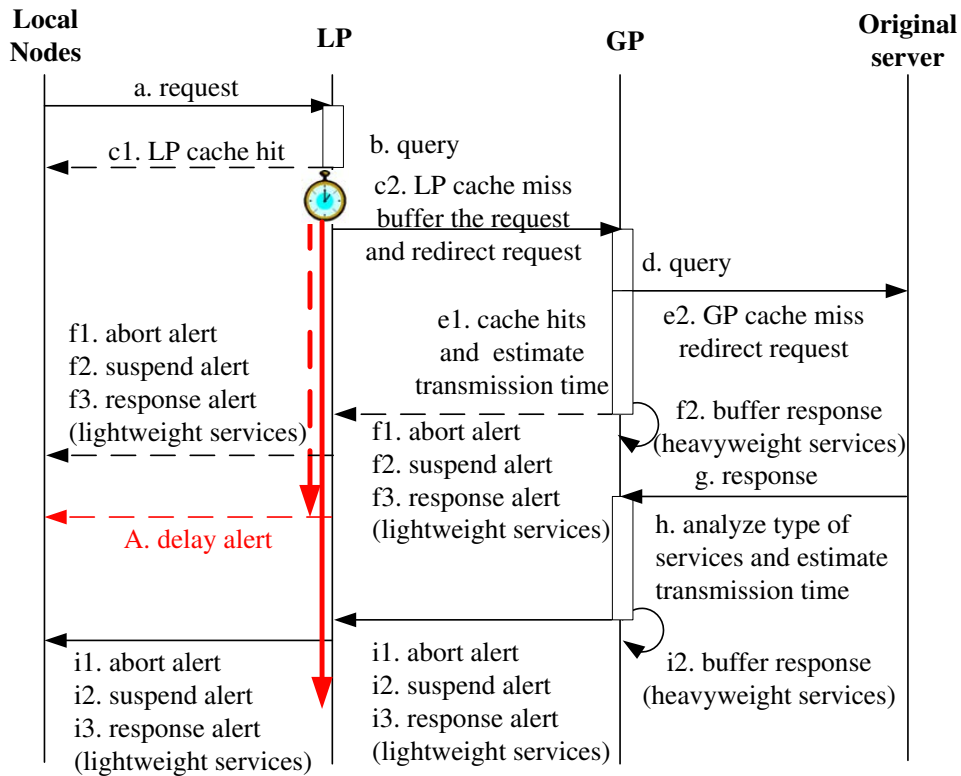


Figure 6.3: Sequence diagram when the MONET is moving (without SPs).

information, the GP will return the information to the LP if the size of the information is not large (process b). If the size of the information is large, the GP will buffer the information and inform the LP to suspend the request. Later, when the MONET arrives at a station, the LP resumes the suspended requests through the low-tier network (process b'). If the GP does not have the requested information, it needs to retrieve information from the designated servers (process c). In the following paragraphs, we will explain the sequence diagrams of the detailed operations first when the MONET is moving between stations and then when the MONET is staying within a station.

(1) Moving between stations

When the MONET is moving between stations, we need to control the usages of the limited bandwidth of high-tier networks. The GP is responsible for suspending heavy-weight requests or aborting the requests that could not be transferred to users before the tolerance time.

As shown in figure 6.3, each request with a cache miss in the LP is associated with a timer to guarantee that users receive the responses or statuses of the requests within a

maximum time delay. When a cache miss occurs for a request in LP, the associated timer may expire before the LP receives the response from the GP. If the timer expires, the LP will send a delay message to inform the user that the request is still under transmission (message A of figure 6.3). Other operations, for the case where the timer does not expire, are described in details as follows.

- a** A user sends a request to the LP.
- b** The LP queries the local cache.
- c** If cache hits, the LP replies the response immediately to the user. If cache misses, the LP redirects the request to the GP, starts the timer, and inserts the request to a waiting queue.
- d** The GP checks if the response is in its local database.
- e** If cache misses, the GP connects to the original server. Otherwise, the GP analyzes the size of the response and determines whether the request is a heavyweight or a lightweight service, and then estimates whether the response could be transferred successfully within the tolerance time, according to the size of the response and the current bandwidth of the communication link.
- f1** If the GP finds that the response, either a heavyweight or lightweight service, cannot possibly be transferred to the user before the tolerance time, it will send an abort message to the LP and thus save the bandwidth for other requests. For the same reason, if the GP finds that the transmission time of a response that is currently under transmission has already exceeded the tolerance time, it will also abort the transmission. When the LP receives the abort message from the GP, it removes the request from the waiting queue and informs the user.
- f2** If the request is a heavyweight service and could be possibly transferred before the tolerance time, the GP will buffer the response and sends a suspend message, instead of the response, with an estimated transmission time to the LP and then to the user. (The response will be sent to the LP later when the MONET is within the

coverage area of the low-tier network.)

f3 If the request is a lightweight service and could be possibly transferred before the tolerance time, the GP will send the response to the LP and then to the user. Once the LP receives the response from the GP, it removes the request from the waiting queue.

g, h, i g, h, i. When the GP receives the response of the cache-miss request from the original server, it caches the response, analyzes the response like step e, and sends the response to the LP like step f.

(2) Staying within a station

When a MONET arrives at a station, the LP can communicate with the GP through the low-tier interface of the MR. Operations performed by the LP and the GP during the staying period are similar to that during the moving period, except step f2 and i2. Since the MONET attaches to the low-tier network, the response for heavyweight services could be transferred to the LP directly. Besides, the LP also resumes the heavyweight requests suspended during the moving period.

2. Load-based scheduling scheme with SPs

When we know the moving track of the MONET, we may add SPs at stations to further shorten the wired-side delay or distribute the load of the GP. In this subsection, we revise the operation procedures for the case where SPs are deployed in the stations.

(1) Moving between stations

Figure 6.4 shows the message flows and operation scenarios of the load-based scheduling scheme with SPs. The message flows are similar to those in the load-based scheduling scheme without SPs. The differences are step f2 and step i2. If the GP classifies the request as a heavyweight service and determines that the response could be transferred to the user before the tolerance time, the GP may send the request and its corresponding response to the SP that the MONET would stop next. When the MONET arrives at the station, the LP can access the response locally from the SP.

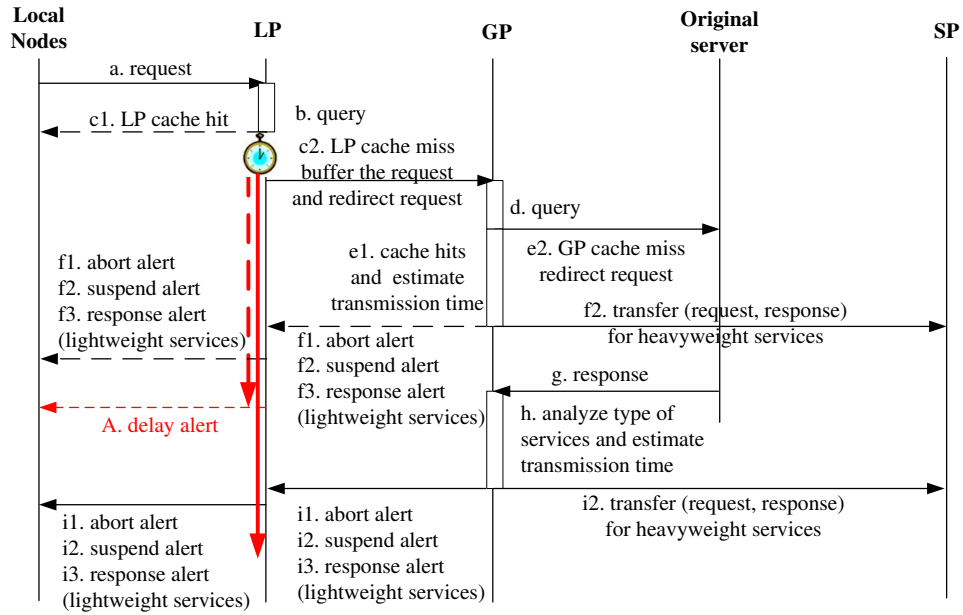
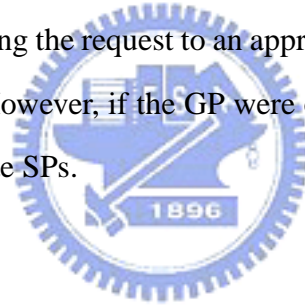


Figure 6.4: Sequence diagram when the mobile network is moving (with SPs).

It should be noted that, in figure 5, we assume the GP fetches the requested information itself, instead of forwarding the request to an appropriate SP and having the SP fetch the information requested. However, if the GP were overloaded, we could have distributed the requests to appropriate SPs.



(2) Staying within a station

When the MONET is staying in the station, operations are the same as those in the scheme without SPs except that the LP can now access information from the SP. Since the GP has already transferred the response to the SP in advance, we can eliminate the wired-side delay between the GP and the SP.

6.1.3 Performance evaluation

We simulate the load-based scheduling scheme without SPs to study the effects of proposed proxy architecture and scheduling scheme on the wireless networks. Without loss of generality, we assume that the MONET moves in a ring of 30 stations with an equal inter-station traversal time and stays at each station for one minute in the simulation. Furthermore, the number of stations a user can traverse follows a uniform distribution with a maximum of 30 stations. Service requests arrive in a Poisson distribution with a mean arrival rate λ , and the

Table 6.1: Simulation Parameters.

Parameters	Description	Default value
λ	Mean request arrival rate (per second)	0.5, 1, 2, K, 20
A	Inter arrival time	Exponential (λ)
S_{Avg}	Mean size of response (Kbytes)	10
S	Size of response (Kbytes)	Geometric ($1/S_{Avg}$)
M	Ratio of moving period to staying period	15
H	Percentage of heavyweight services	0.4
L	Percentage of the low-tier bandwidth that lightweight services can use during staying periods	0.1
r	Hit ratio of LP	0.5
R	Hit ratio of GP	0.5
T_{light}	Tolerance time of lightweight services (s)	10
T_{heavy}	Tolerance time of heavyweight services (s)	1000
B_{low}	Bandwidth of low-tier networks (Kbps)	5000
B_{high}	Bandwidth of high-tier networks (Kbps)	50
I	Internet Access (ms/Kbyte)	1

data size of responses follows a geometric distribution with a mean S_{Avg} . Other parameters are summarized in Table 6.1.

We show the completion rate and the average waiting time for both lightweight and heavyweight services. The completion rate is defined as the probability that a user can receive response completely before the tolerance time and the waiting time is the elapsed time from the time a request is issued to the time the corresponding response is received completely.

1. The effects of the hierarchical-proxy architecture in two-tier networks

Figure 6.5 presents the completion rates in two-tier and single high-tier networks with different hit ratios of the LP, denoted as r . In general, the completion rate increases with r but decreases as λ increases. When the arrivals saturate the wireless bandwidth between the LP and the GP, the completion rate reaches its lower bound. The lower bound of the completion rate equals r for both lightweight and heavyweight services. Furthermore, the completion rate of service requests in two-tier MONETs reach the lower bound much later than that in single high-tier MONETs. The reason of the delay is that the two-tier MONET offers much higher wireless bandwidth than the single high-tier MONET does. Moreover, because we assume a higher tolerance time for the heavyweight services, the completion rates of the heavyweight

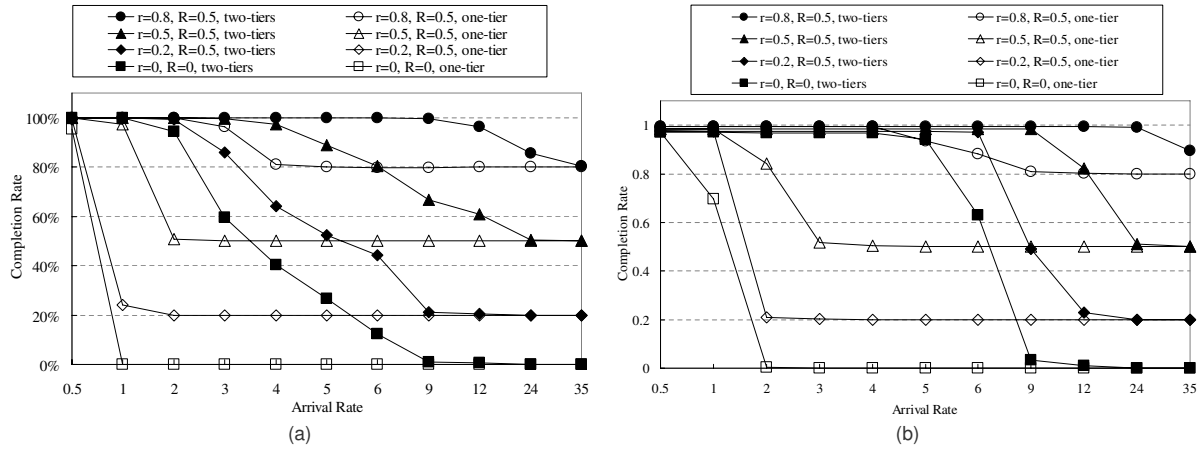


Figure 6.5: Completion Rates in two-tier and single high-tier networks.

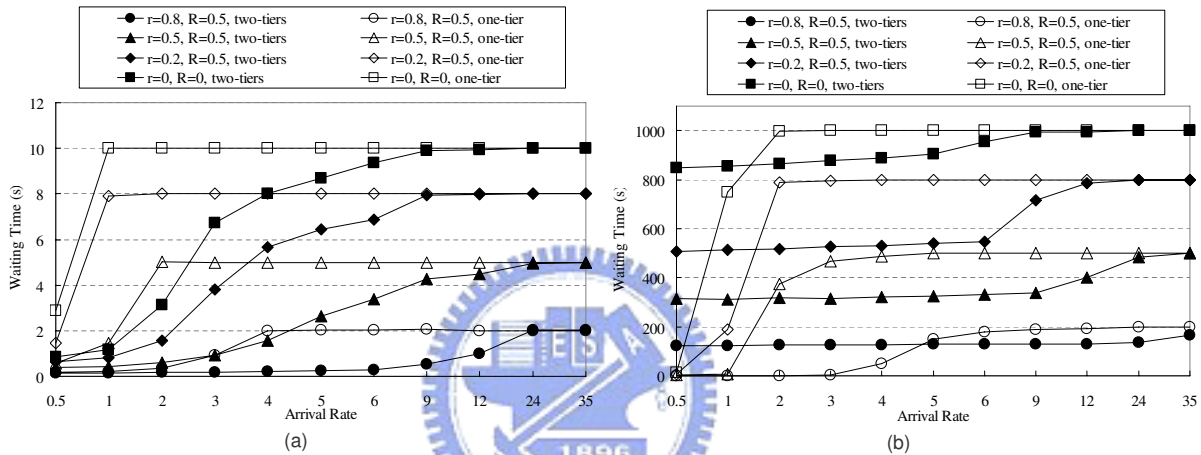


Figure 6.6: Average waiting times in two-tier and single high-tier networks.

services start to degrade much later than that of the lightweight services do.

Similar to Figure 6.5, Figure 6.6 presents the effect on average waiting time. The average waiting time decreases as r increases and increases with λ . Similar to the completion rates, the average waiting times also reach the upper bound when the traffic saturate the wireless bandwidth. The upper bound of average waiting time (W_u) can be estimated as follows.

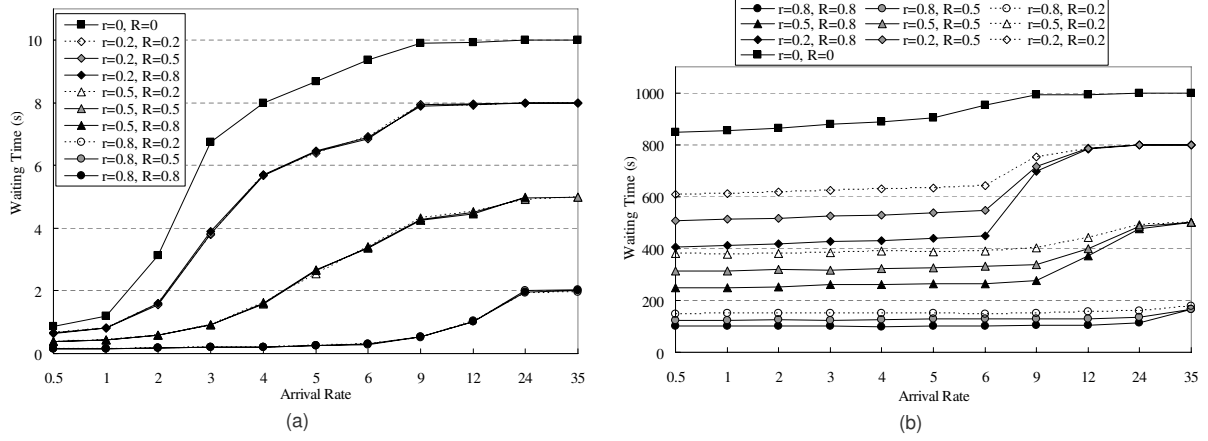


Figure 6.7: Average waiting times with different hit ratios of the GP.

Lightweight services:

$$W_u = (1 - r) \times T_{light} + r \times \left(\sum_{k=1}^S \frac{P\{X = k\}}{P\{X \leq S\}} \times k \right) \times 8/B_{low}$$

Heavyweight services:

$$W_u = (1 - r) \times T_{heavy} + r \times \left(\sum_{k=S+1}^{\infty} \frac{P\{X = k\}}{P\{X \leq S\}} \times k \right) \times 8/B_{low}$$

Again, the average waiting time in two-tier MONETs reaches the upper bound much later than that of single high-tier MONETs. However, it should be noted that in our load-based scheduling scheme, the heavyweight services could only be transmitted over the low-tier network when the MONET stays within a station. Therefore, the average waiting times of heavyweight services in two-tier MONETs are larger than that in single high-tier MONETs when the arrival rate is small.

Figure 6.5 and Figure 6.6 have shown the effects of the LP, and here Figure 6.7 illustrates the effects of the GP. When a request results in a cache miss at the GP, the GP needs to fetch the corresponding response from the original server. Fetching Information across the Internet will result in an extra delay. However, if the size of the information requested is small, the delay may be negligible. Therefore, as shown in Figure 6.7 (a), the effects of the hit ratios of the GP (R) are not obvious for the lightweight services, and there is the negative correlation between the waiting times and R for heavyweight services. On the contrary, as R increases

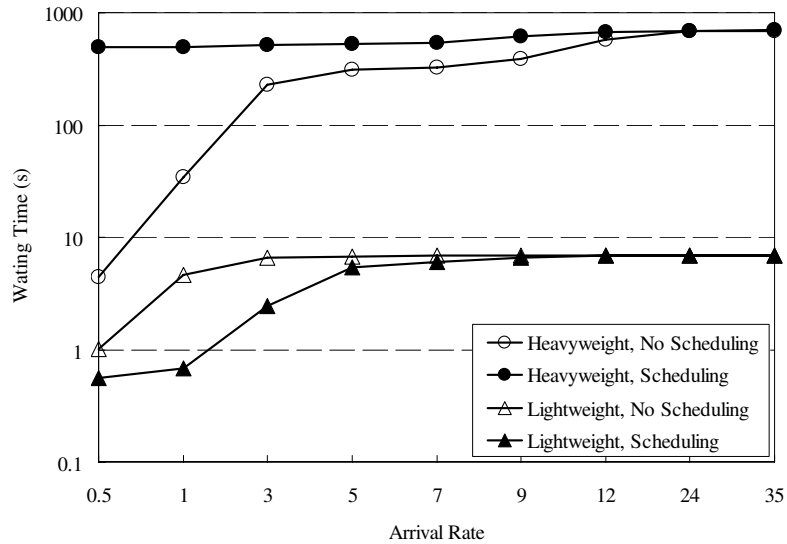


Figure 6.8: Average waiting times with and without scheduling.

the waiting time decreases for the heavyweight services.

2. The effects of the load-based scheduling scheme

We will analyze the effects of the proposed load-based scheduling scheme in this subsection. Since the effects of hierarchical proxy and two-tier networks have already been studied in the last subsection, we will present only the simulation results under two-tier wireless networks with both GP and LPs in this subsection.

Figure 6.8 illustrates the effects of the load-based scheduling scheme on the average waiting time. The load-based scheme could reduce the average waiting times of the lightweight services, but will increase the average waiting times of the heavyweight services. The reason is we allow the heavyweight services only being transferred through the low-tier networks. Therefore, requests of heavyweight services issues between stations cannot be served by the proxies until MONETs arrive at stations, and thus the average waiting time of heavyweight services increases.

Figure 6.9 illustrates the effects on the completion rate of the load-based scheduling scheme. The load-based scheduling scheme could significantly increase the completion rate of lightweight services with a slightly decrease in that of heavyweight services. In other words, the load-based scheduling scheme could significantly increase the completion rate of lightweight services and still maintain about the same completion rates of heavyweight services.

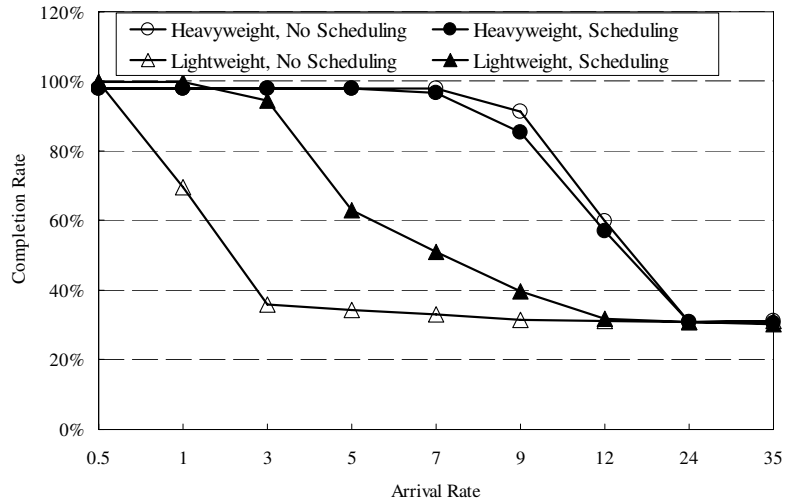


Figure 6.9: Completion rates with and without scheduling.

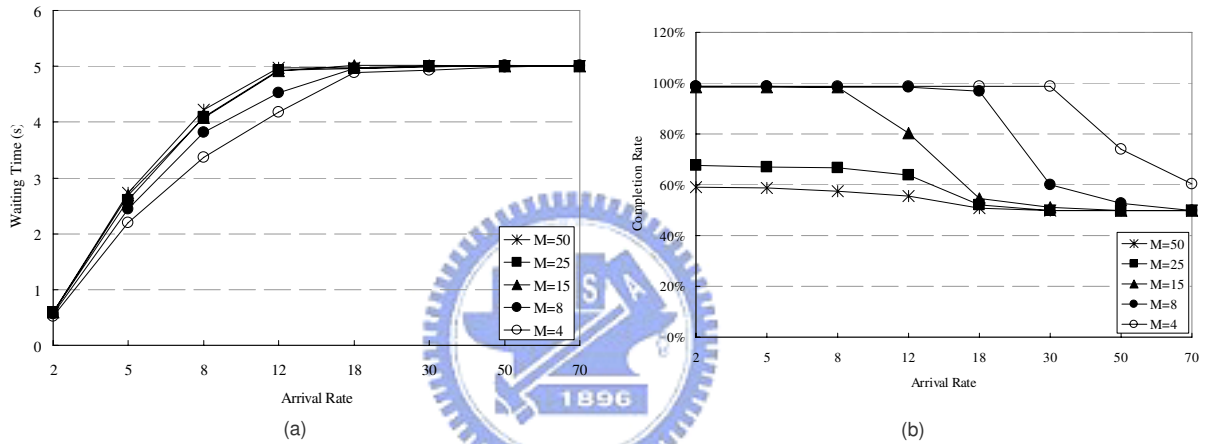


Figure 6.10: Performance for different ratios of moving periods to staying periods.

3. The effects of other environment factors

We would also like to examine the effects of other environment factors. Because of the interactive natures of lightweight services and the just-in-time natures of heavyweight services, we will focus our discussion on the average waiting times for lightweight services and the completion rates for heavyweight services in the following analyses.

Figure 6.10 shows the effects of the ratios of moving periods to staying periods of the MONET, denoted as M . When M is small, the MONET stays at the station relatively long. In our simulation, we reserve 10% of the low-tier bandwidth, which is 10 times of the high-tier bandwidth, to the lightweight services. Therefore, when M is smaller, the average bandwidth allocated to the lightweight services is higher, and thus the average waiting times of the lightweight services are shorter as shown in Figure 6.10 (a). On the other hand, the heavy-

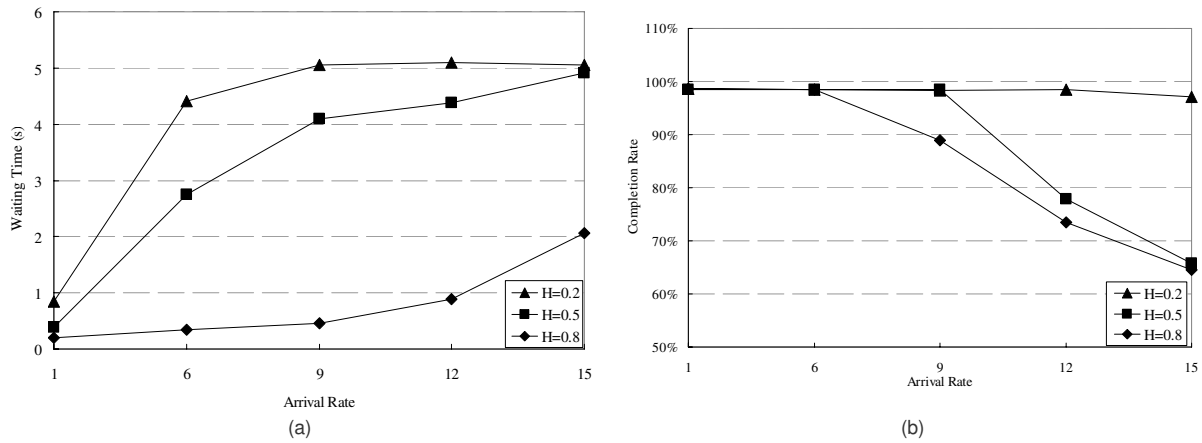


Figure 6.11: Performance for different percentages of heavyweight services.

weight services can only be transmitted during the staying period. An increase in M will result in a decrease in the completion rates of the heavyweight services, as shown in Figure 6.10 (b). If M is so large that the duration of one moving period exceeds the tolerance time of heavyweight services, the request of a heavyweight service can only be served in one station at most. Therefore, when M is large, the completion rates are low even when the arrival traffic is light.

Figure 6.11 shows the effects of the percentage of the heavyweight services, denoted as H . When H is large, most requests are heavyweight services, and the bandwidth of high-tier networks is sufficient enough to serve lightweight services. Therefore, when H increases, the average waiting times of lightweight services decrease, as shown in Figure 6.11 (a). On the contrary, as shown in Figure 6.11 (b), the completion rates of heavyweight services decrease as H increases because more heavyweight services will share the bandwidth of low-tier networks when H is high.

Figure 6.12 presents the effects of L , which is the percentage of low-tier network bandwidth that could be used by the lightweight services. As shown in Figure 6.12 (a), the higher L is, the more the bandwidth lightweight services can use, and thus the shorter the average waiting time of lightweight services is. However, when L is high, the bandwidth allocated for heavyweight services is relatively low. Therefore, as shown in Figure 6.12 (b), the completion rates of heavyweight services drop when L increases.

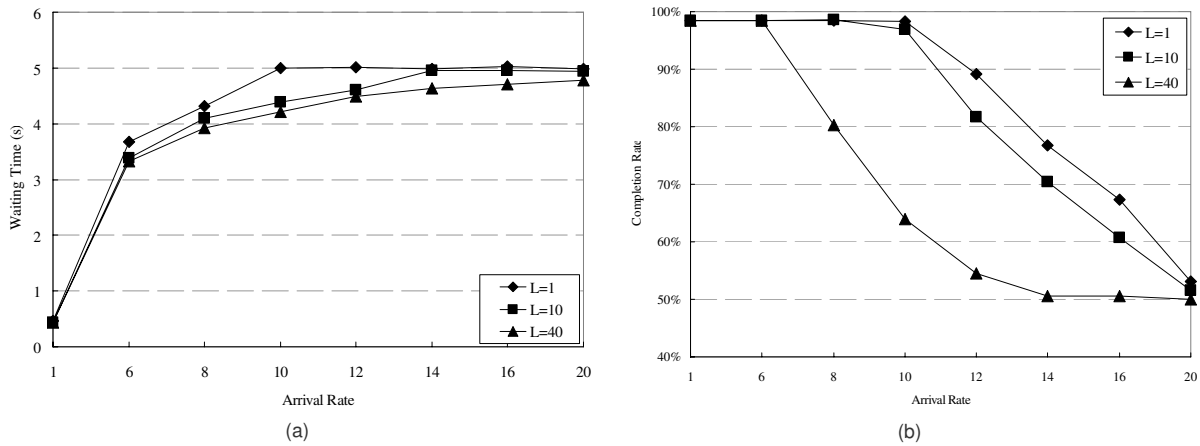


Figure 6.12: Performance for different percentages of low-tier bandwidth for lightweight services.

6.2 Hierarchical personal information management

In the past decades, memory hierarchy runs very successfully in managing data in computer systems. In computer systems, hard disks play as permanent storage devices to store large amount of information, but they are limited in the access speed. Besides, not all information is required by Central Processing Unit (CPU) in a time, and thus only necessary information will be copied to the main memory, which has higher access speed than hard disks have. In order to further increase accessing speed, people add caches closer to or even in the CPU. In a summary, the closer a memory is to a CPU, the higher is its access speed and cost per bit, but the less is its storage space. Based on the concept of memory hierarchy, computers can give advantages for both access speed and data storage space.

Personal information such as e-mails, schedules, address books, memoranda or business cards is another sort of huge amount of data for modern people to manage. With the advance of technologies, people now may manage their personal information via various devices, such as cellular phones, Personal Digital Assistants (PDAs), personal computers, or even information servers directly. The roles and behaviors of these devices are very similar to those storage components in computer systems. Take Figure 6.13 for example, information servers located in the Internet behave like hard disks to store huge amount of permanent information. Every user is like a program and copies his own personal information to a personal computer. However, it is not always possible for people to have their personal computers in handy. To en-

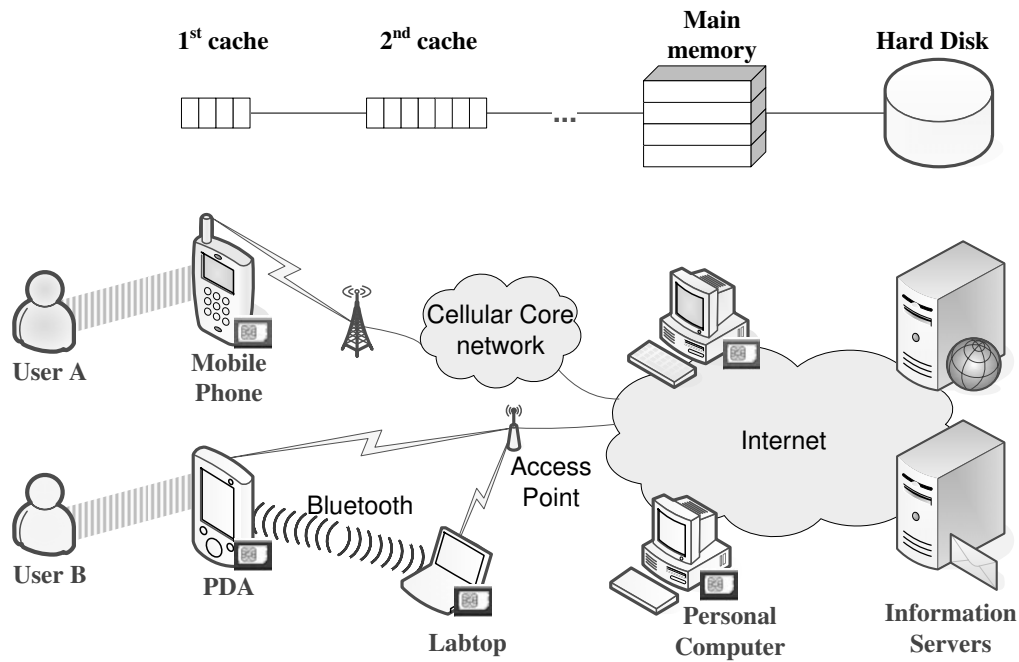


Figure 6.13: Example of four-layered repositories.

hance the portability, people may copy their information to a laptop, a PDA or even a cellular phone. It seems that we can apply the successful experience of memory hierarchy to manage personal information for multiple devices to gain both advantages of portability and storage space. Unfortunately, the characteristics of personal information are not totally same as what of data in computer systems. As a result, the policies applied in memory hierarchy can not be applied to information hierarchy directly.

The success of a memory hierarchy lies on the program behavior. According to the temporal and spatial localities of accessing patterns, computers can decide which data should be copied to caches in advance or which data should be replaced in caches. However, users behave different while accessing personal information, and various services may also influence their accessing behaviors of personal information. Therefore, it is not easy to decide which information should be transferred to or be replaced in mobile devices.

Personal information is more complicated than each data entry in computer systems. In computer systems, each data entry has a unique address, which can lead to a single data entry no matter the data is in the hard disks, main memory, or caches. Nevertheless, an identity of personal information usually corresponds to a group of data. For example, the identity of an e-mail corresponds to at least the sender, title, date, content of a message. We do not need

to treat all fields of information as an atomic data, since they are not equally important and necessary to users especially when users use limited-resource mobile devices. Consequently, different mechanisms are required to manage compound personal information.

In this section, we propose architecture of *Personal Information Hierarchy (PIH)* [64] and access policies according to the characteristics of personal information services. Since different users or services may lead to various accessing behaviors, the proposed PIH manages personal information based on user and service preferences. In addition, a single row of personal information consists of several fields of information, and thus the proposed PIH also performs abstraction to extract important fields of personal information. Only important fields of information would be sent to mobile devices to reduce the size of information transmitted in the scarce wireless bandwidth and stored in the limited memory space of mobile devices.

6.2.1 Personal information hierarchy

Like the memory hierarchy, PIH consists of layered repositories (Figure 1). Memory hierarchy emphasizes the accessing speed from CPU, and the PIH similarly respects the accessing speed from users. Users can carry mobile devices having higher portability to increase the access speed, but such devices are usually limited in the processing power and storage space. Information servers on the Internet can make up the storage requirement. In a summary, in the PIH, a preceding layer starting from mobile devices to the information servers has higher portability but less storage space and computing power than a successive layer.

Mobile devices play roles as caches in the lowest level in the PIH. Mobile devices can provide high access speed but poor storage space to users. Once some information is missing in mobile devices, it needs to be retrieved from the Internet. Without loss of the portability, mobile devices in the PIH need to have the ability of wireless communication, such as Wireless LAN (WLAN) or cellular interface like General Packet Radio Service (GPRS) or Personal Handy-phone System (PHS). Mobile devices can also have some other short range wireless transmission interface, such as Bluetooth or infrared rays, to communicate with other mobile devices. For example, user B in the Figure 1 can connect to a laptop by Bluetooth to retrieve data when information is missing in his PDA. The multiple layers of mobile devices are like multiple caches, and they can reduce the latency of establishing connection to the servers in

the Internet when each miss occurs.

The personal computer and information servers located in the Internet are the backups of mobile devices, and they act like the main memory and hard disks in the memory hierarchy respectively. There may be many information servers existing on the Internet to provide various kinds of personal information services, and thus the personal information of a certain user may exist in several places on the Internet. The personal computer behaves like a bridge. On the one hand, it gathers and stores information from information servers, and provides information to mobile devices whenever information missing on the other hand.

Memory card in the PIH is an additional component to perform authentication and personalization. Without doubt, personal information is very important and private to the owner, and thus authentication is required in the PIH. However, the inefficient input interfaces of mobile devices might harm the pleasures of using services if users are asked for authentication each time they access information. We provide an alternative for users to store their account information in a memory card. In that case, users can just plug the memory card into the devices and the authentication will be performed automatically. Personalization is the other objective of memory card. Users can set their personal preferences in the memory card. The personal preferences, such as application environment or important contacts, can not only help to automatically configure environments of a new device, but can also help the PIH provide personalization information services to users. As a result, we add a new component of memory card in the PIH to enhance the efficiency of personal information services.

6.2.2 Information accessing policies

In this section, we describe the proposed policies for the PIH. A row of personal information usually consists of several fields, and all fields are not equally important or necessary to users. In order to avoid sending unnecessary information to the mobile devices, we divide a row of information into three categories: primary, content, and ignored information, according to their importance. We first assign a weight value for each field to represent its importance, and then sort all fields by the value of weight like Figure 6.14. The most important fields belong to primary information; the least important fields are ignored information; and the rest fields are content information. The weight value of each field and the boundaries between three

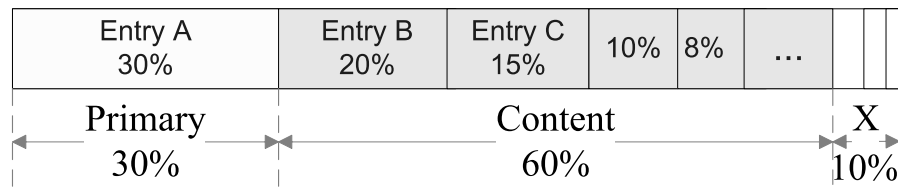


Figure 6.14: Example of information division.

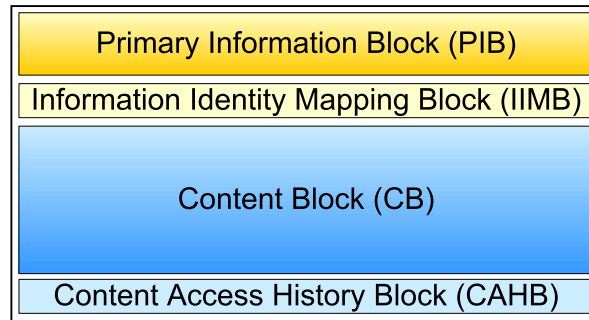


Figure 6.15: Memory blocks in mobile devices.

categories of information can be set by users or learned by systems automatically for different services and devices.

We transfer only primary and content information to mobile devices, and Figure 6.15 illustrates the memory blocks in a mobile device to store information. *Primary Information Block (PIB)* and *Content Information Block (CIB)* store primary and content information respectively. *Information Identity Mapping Block (IIMB)* records the mapping of global and local identity of information, and the *Content Access History Block (CAHB)* records the access histories of content information. In the following, we describe the reading, writing, and replacement policies for PIH in the following subsections.

1. Two-stage Information Reading Policies

A user is not like a program that exactly knows which data entries are required during the execution. For example, a user may not be aware of some new mails sent to the mail server, or a new schedule is arranged in the calendar by the secretary. In case the user has no knowledge of which information could be accessed, there are both dominant and recessive reading requests in the PIH. When the user knows which information he wants, he can issue a dominant reading request, otherwise he can just send a recessive reading request. A dominant request can also be embedded with a recessive request to receive new information automatically.

We allow users to specify the conditions of recessive reading requests in their personal preferences to avoid setting conditions each time. For example, users may specify to receive the business cards of certain groups or the mails sent in three daysK etc. The personal preferences can help the personal computer filter out unnecessary information to be sent to the mobile devices.

When a user uses a new mobile device, he could send a recessive request first. The mobile device then forwards the recessive request along with the authentication and preference information to his personal computer. The personal computer authenticates the user and asks for new information from all information servers. Upon receiving the response from servers, the personal computer first filters the information according to the preferences as mentioned earlier, and transfers the filtered information to the mobile device in two stages. In the first stage, the personal computer will send the primary information to give the user a brief list of his personal information. If the user is interested in a certain row, he can issue a dominant reading request to acquire the content information in the second stage.

When a user issues a dominant reading request in his mobile device, the mobile device first checks if the content information exists in the CIB. If a miss occurs in the CIB, the request will be forwarded to the personal computer or other mobile devices to retrieve the required content information.

2. The writing policy

There are two main writing policies in the memory hierarchy. In the PIH, we adopt the policy of writing back in general and writing through only when users require. Since the wireless bandwidth is expensive, it will cost a lot if we always keep updating information on the server. In addition, users usually use only one device at the same time, and there is no necessity to update information immediately. Therefore, we update information only when users are going to close the application or under user's requirement in case of emergency.

3. The replacement policy

In this subsection, we discuss about the replacement policies of both PIB and CIB in the PIH. Although many replacement schemes have been proposed for the memory hierarchy

of computer systems [29, 48, 41, 7], they are not directly applicable to the PIH because of different access behaviors.

When there is not enough space in the PIB to store primary information, we adopt sequential replacement since users usually browse information list sequentially. Specifically, when the rows of information exceed the number of PIB can contain, the first beginning rows of information will be replaced and vice versa. For example, if PIB can store a maximum of a hundred entries and there are 110 rows of information, the first ten rows will be replaced by the information with the number from 101 to 110. The mapping of local and global identities of replaced primary information will be kept in the IIMB. When the user rolls back to the beginning rows of information, the mobile device will send a recessive request by specifying the global identities of information to the personal computer to retrieve primary information back, and the last ten rows will be replaced.

The replacement of CIB is more complicated than that of PIB because users do not access content information sequentially. In addition, the size of content information is bigger than primary information, and inefficient replacement policies may seriously waste wireless bandwidth and money. However, mobile devices have limited processing power, and they can not execute a too complicated replacement policy. Therefore, we propose a user-assisted replacement mechanism which can adjust the replacement policy according to different services and devices.

A user is the one who is the most conscious of the importance of each row of his personal information. The user can mark a row of information if he is likely to access it again. Unmarked information will have a lower priority and will be removed earlier than marked information when the CIB does not have enough space to accommodate new content information.

For the most limited-resource mobile devices like cellular phones, user decision is the only criterion to make the decision of replacement. For other more powerful mobile devices like PDAs, we can add other criteria to help mobile devices make a more precise decision of replacement. The more powerful a mobile device is, the more criteria it can apply for the replacement. The criteria should be added in the order of the importance weight value of each

```

while(free space is not enough){
  while (there are unmarked mails and free space is not enough){
    victim_sender = select_victim_sender();
    while (there are unmarked mails whose sender is victim_sender
      and free space is not enough ){
      victim_mail = select_victim_mail (victim_sender, "unmarked");
      delete(victim_mail);
      add_free_space(victim_mail);
    }
  }

  victim_sender = select_victim_sender();
  while (there are marked mails whose sender is victim_sender
    and free space is not enough ){
    victim_mail = select_victim_mail (victim_sender, "marked");
    delete(victim_mail);
    add_free_space(victim_mail);
  }
}

```

Figure 6.16: Algorithm of three-level replacement policy by taking e-mail for example.

field we mentioned in the beginning of Section 3. For example, if the most important entry of e-mail is the sender, sender will be the second criterion after user decision. The mobile device needs to record the access history of each sender in the CAHB. The access history includes the last access timestamp and the access count of content information sent from the certain sender. These two values can obtain a replacement weight value for each sender by

$$RWeightValue(Sender_i) = \alpha \cdot TimeStamp(Sender_i) + (1 - \alpha) \cdot AccessCount(Sender_i).$$

Figure 6.16 is an algorithm of replacement in e-mail services if we apply two criteria, user decision and sender.

6.2.3 Service adaptation strategies

There are lots of personal information services, and they may have different characteristics. In this section, we give an example of how to divide a row of information into primary, content, and ignored parts for three kinds of personal information services, which are e-mail, calendar, and business cards. Former studies [52, 30, 18] show that how people access their personal information is highly correlated with the owner and the time of information. We can briefly conclude our observations below and follow these observations to list examples of the abstraction results in Table 6.2. Since the storage space of PDAs is larger than that of cell phones,

Table 6.2: Example of abstraction for e-mail, calendar, and business card.

	E-mail		Calendar		Business card	
	Cellular phone	PDA	Cellular phone	PDA	Cellular phone	PDA
Primary information	Sender, Title	Sender, Title, Date/time	Participant, Date/time	Participant, Date/time, subject, place	Owner, Phone	Owner, Phone, Email addr.
Content information	Date/Time, Mail body, Title of attachment	Mail body, Attachment	Subject, Place, Description	Description	Email addr, Affiliation, H/O addr., URL, Instant msg	Affiliation, H/O addr., URL, Instant msg
Ignored information	Version, Path, ...	Version, Path, ...	Reminder, ...	Reminder, ...	Birthday, Nickname, ...	Birthday, Nickname, ...

we can put more information in primary parts to decrease the probability for users to issue dominant requests for content information.

- **Observation a.** Important contacts usually have higher longevity and reciprocity, and are more recent than unimportant contacts.
- **Observation b.** Important contacts usually are in the same affiliation.
- **Observation c.** Users’ decisions to delete e-mails usually depend on who is the sender, such as the number of previous mails sent by the sender and the interaction frequency with the sender.

6.2.4 Performance evaluation

We evaluate the performance based on our prototype in this section. We first study the number of mails an email system can store in a mobile device with or without applying PIH.

In this section, we demonstrate the effectiveness of the HPIM scheme on the number of mails available to a user. As mentioned earlier, the memory space of an HPIM e-mail system is divided into four blocks: IIMB, PIB, CB, and CAHB. An HPIM device keeps only the *primary information* of mails, instead of storing the *full header* of each mail, as an ordinary

Table 6.3: Entry sizes of IIMB, PIB and CAHB of a cellular phone.

memory block	entry size
IIMB	8 bytes
PIB	40 bytes
CAHB	21 bytes
CB	Flexible

PIM e-mail system does. Therefore it has more space for PIB, IIMB, and CAHB to fulfill the virtual memory concept of HPIM. In an HPIM mail system, the number of CAHB entries is the same as the number of mails having their contents stored in CB. However PIB and IIMB can have as many entries if there is still available spare memory available. The total entry number of PIB and IIMB is the number of mails accessible by an HPIM user. In order to focus on the effects of PIB and IIMB memory allocation on the perceivable mail numbers, we assume the number of mail contents stored in an HPIM mobile device is the same as that in an ordinary PIM device in the following analysis. Furthermore, the average size of a mail is set as 40K bytes with a full header of 1K bytes. The entry sizes of PIB, IIMB, and CAHB of an HPIM mobile phone are 40, 8, and 21 bytes, respectively, as shown in Table 6.3.

We first simply set the size of IIMB as zero and study the effects of PIB by varying the memory sizes reserved for the e-mail system. When we set IIMB to zero, and we can calculate the number of mails accessible by an HPIM e-mail system as follows.

$$N_v = \frac{\text{Memory Space} - (\text{Avg. size of content} \times N_{CB} + \text{Entry size of CAHB} \times N_{CB})}{\text{Entry size of PIB}}$$

where N_v is the virtual number of mails accessible and N_{CB} the number of mail contents stored in an HPIM e-mail system.

Figure 6.17 shows the number of mails stored in an ordinary PIM e-mail system and the number of mails accessible in an HPIM e-mail system with various sizes of memory space. In Figure 6.17 (a), we set the size of IIMB as zero, and we can observe that the number of mails accessible in an HPIM e-mail system is much larger than the number of mails stored in an ordinary PIM system. In figure 6.17 (b), we study N_v under different percentage of IIMB and PIB. We can observe that N_v increases as the percentage of IIMB increases since the entry size of IIMB is much smaller than that of PIB. However, this may also influence the hit ratios.

Figure 6.18 gives us an idea of how the percentage of IIMB influences miss ratios of both

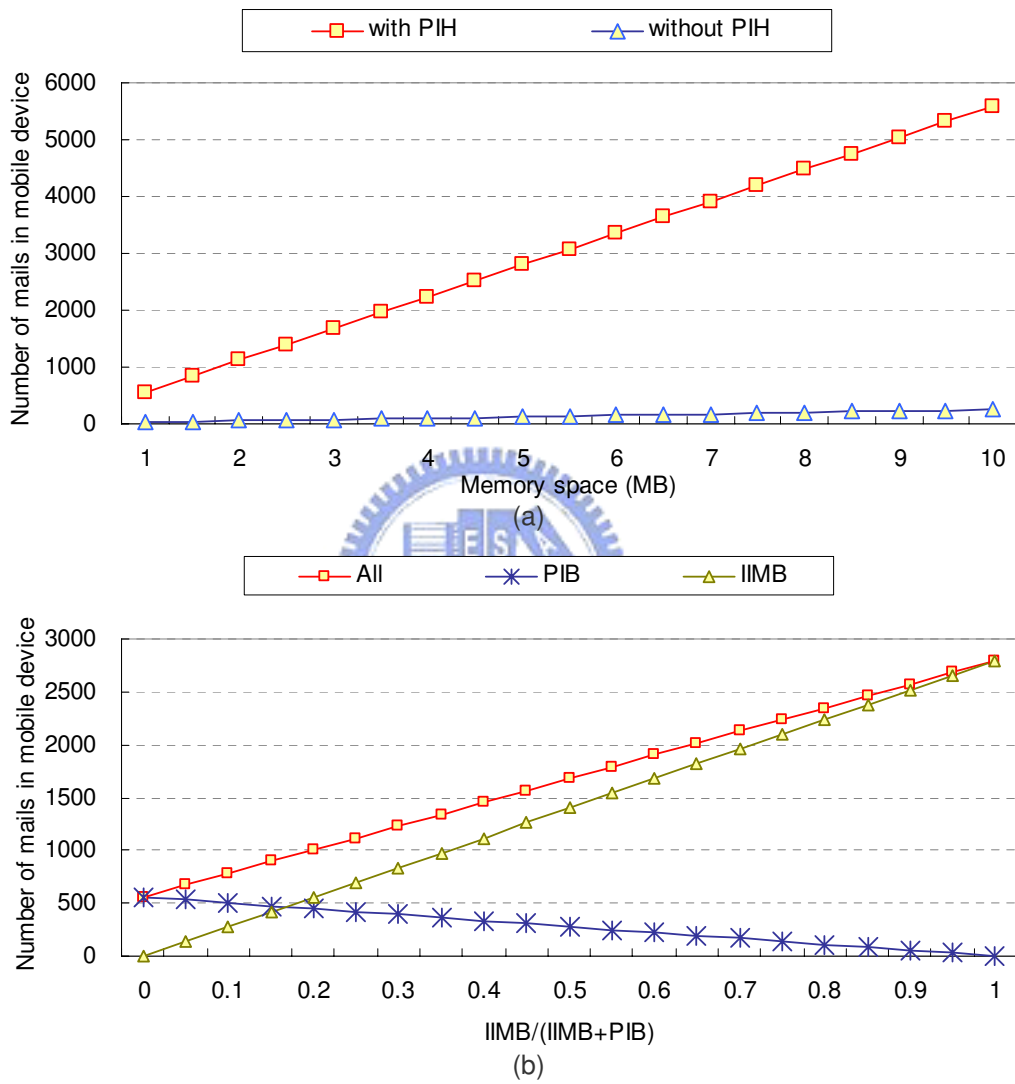


Figure 6.17: Effect of information hierarchy on number of mails with different memory space sizes.

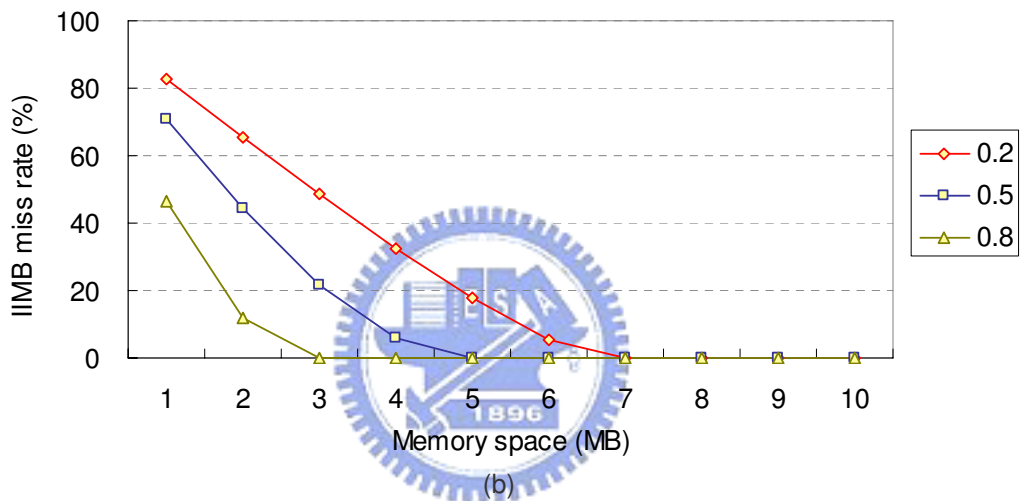
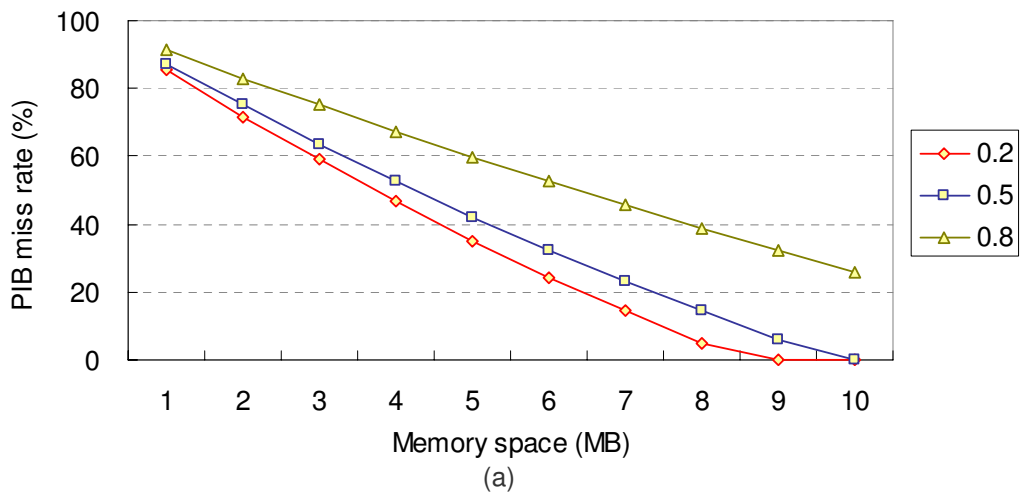


Figure 6.18: Effect of the size of IIMB on miss rates of PIB and IIMB.

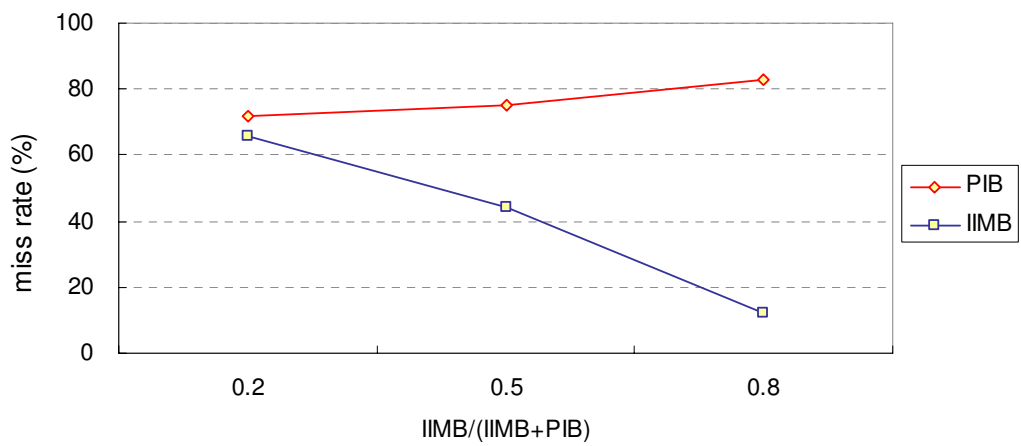


Figure 6.19: Miss rates of PIB and IIMB with different percentage of the size of memory blocks and fixed number of mails.

IIMB and PIB. In general, when the memory space increases, miss rate of both IIMB and PIB decreases. If we increase the portion of IIMB, it will cause a decrement of IIMB miss rate but an increment of PIB miss rate. However, we can find in Figure 6.18 (b) that if the memory space is big enough (7 MB in Figure for example), the miss rate of IIMB is zero. Therefore, we can arrange larger size of memory space for PIB when memory space is large enough to decrease the miss rate of PIB and thus decrease information access time.

We can also observe from Figure 6.19 that if we spare some memory space from PIB to IIMB, it will slightly increase the miss rate of PIB but can improve lots of hit ratio in IIMB. The reason is because the entry size of IIMB is much smaller than that of PIB. Therefore, a small sacrifice of size of PIB will greatly improve the hit ratio of IIMB and increase the number of virtual mails in mobile devices.

Besides the size of IIMB and PIB may influence miss rate and access time, different information access behaviors may also influence the performance of PIM system. We generate four access behaviors to study how they effect on the performance. The First one is *random access*, which access information randomly. The second one is *Sender locality access*, in which users have higher probability to access some information sent by particular group of senders. The third one is *Time locality access*, in which users have higher probability to access newer information. The last one is *Sender and time locality*, which combines characteristics of both time locality access and sender locality access behaviors.

Figure 6.20 illustrates the results how different access behaviors influence the miss rate of IIMB, PIB, and CB. Figure 6.20 (a) and (b) are miss rates of PIB and IIMB respectively. Since we apply sequential replacement strategy for both PIB and IIMB, time is the most critical factor to decide which information should be replaced. As a result, access behaviors with time locality have lower miss rate in PIB and IIMB. On the contrary, user-assisted replacement mechanism is applied for CB and it considers all three factors, user decision, time, and sender. Therefore, access behaviors with sender locality have lower miss rates in CB and the fourth access behavior, Sender and time locality access, has lowest miss rate while memory space increases.

Number of mails in the information server may also influence the hit ratio of PIB and

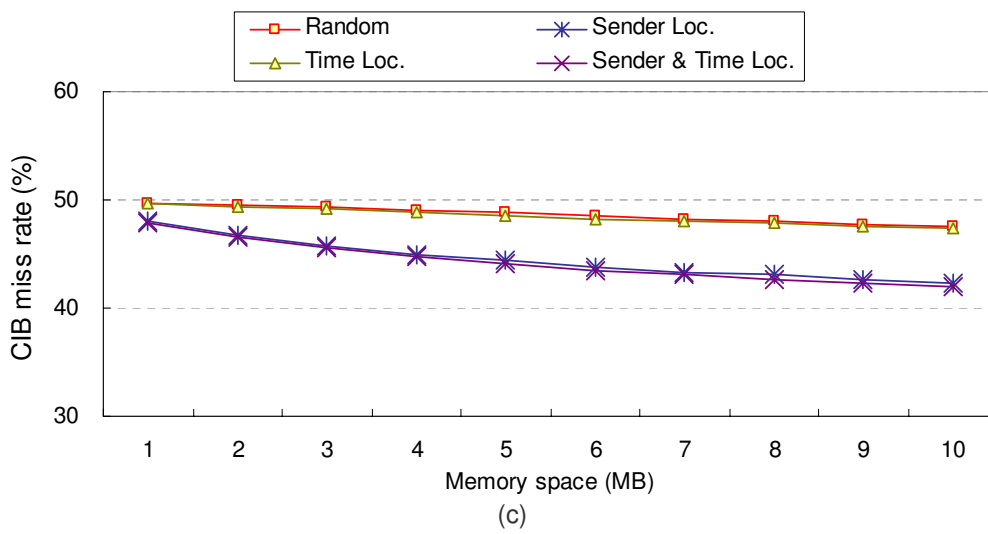
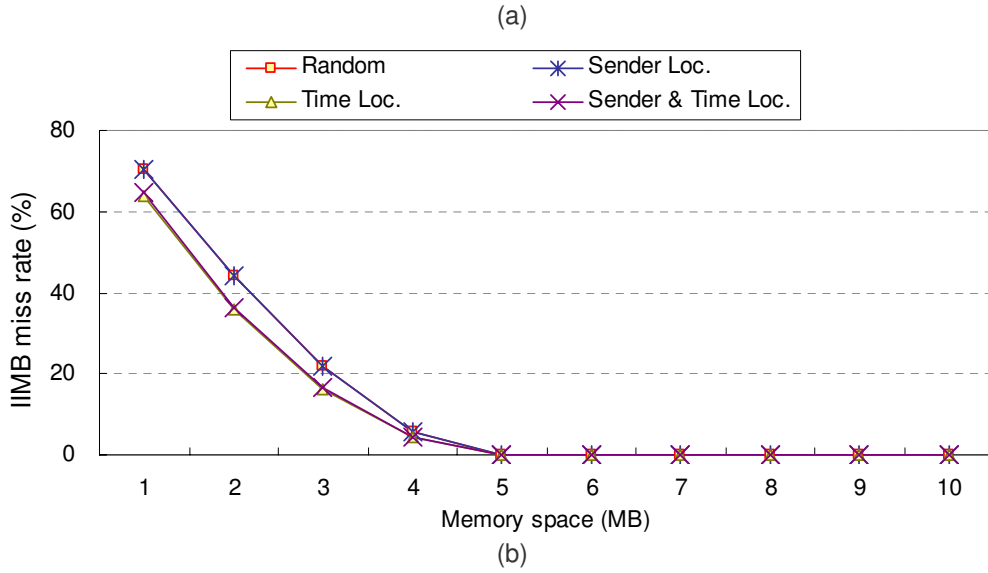
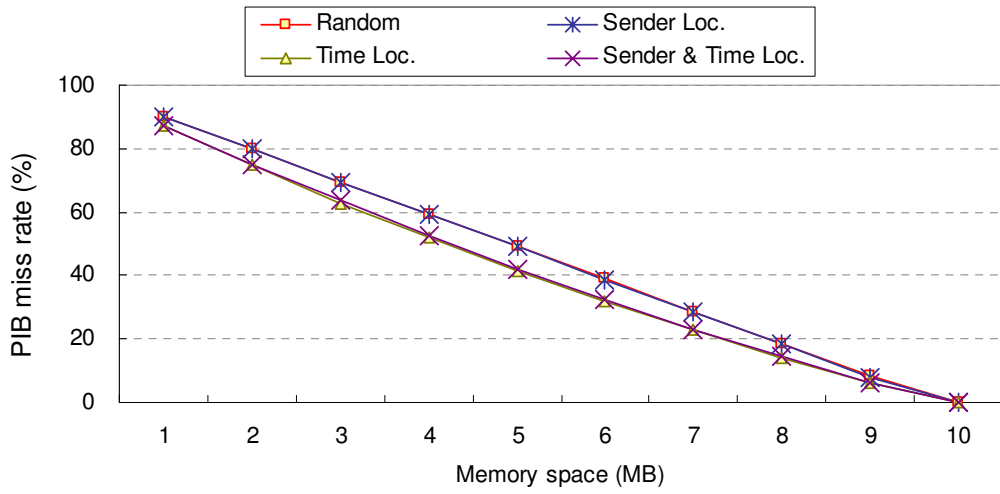
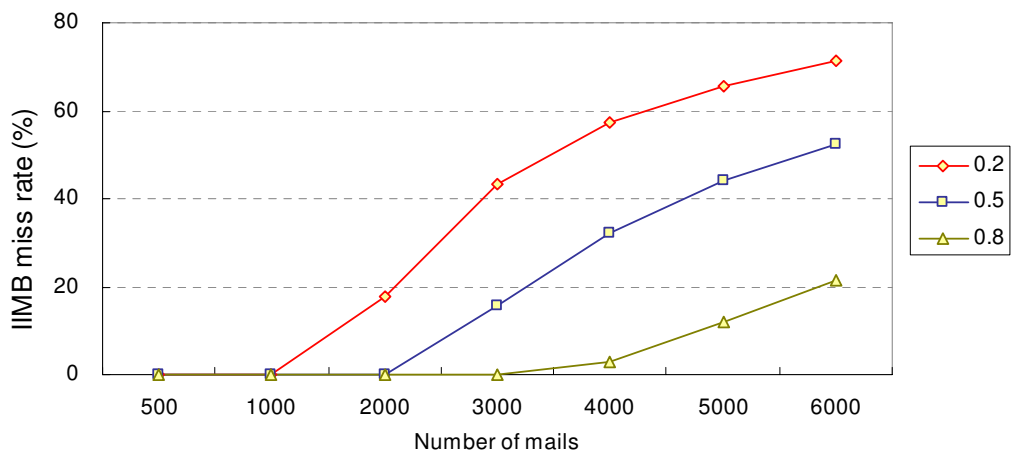
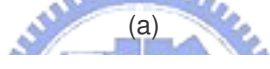
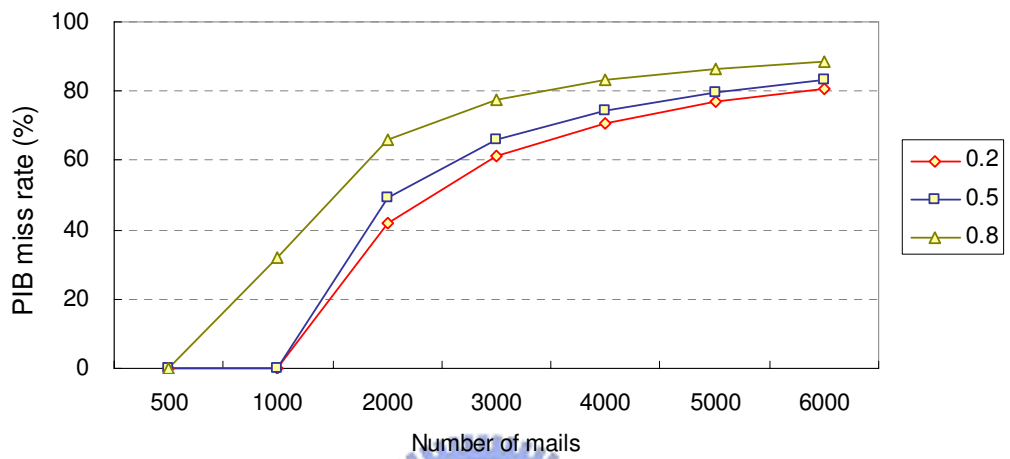


Figure 6.20: Effect of different access behaviors on miss rates of PIB , IIMB, and CIB.



(b)

Figure 6.21: Miss rates of PIB and IIMB with different percentage of the size of memory blocks.

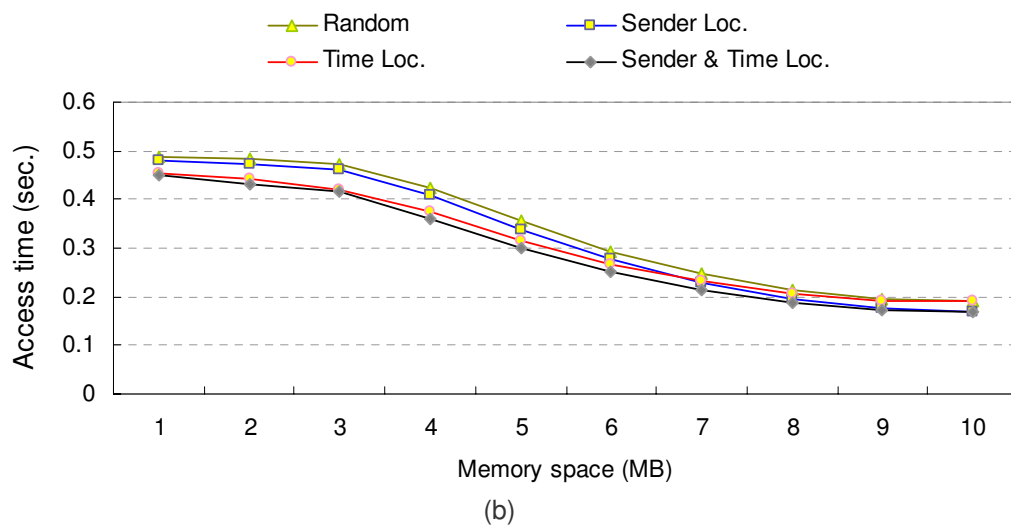
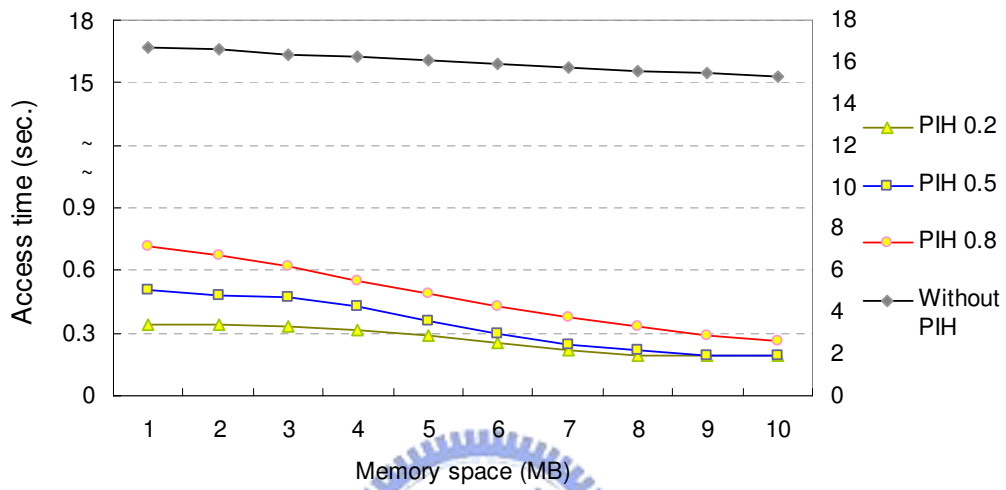
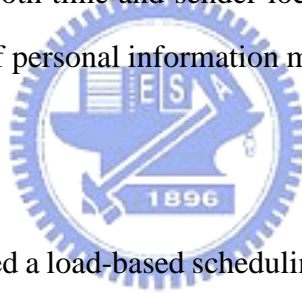


Figure 6.22: Effect on average access time.

IIMB. When there are more mails in the information server, the probability of replacement increases since the memory space of mobile device is fixed and the number of physical mails can be stored is also limited. Figure 6.21 shows the effect of number of mails in the server.

The most important performance result we need to study is average access time of information. Figure 6.22 demonstrates the effects of IIMB size and access behaviors on average access time. In Figure 6.22(a), we can observe that users need to wait for at least ten seconds for accessing a mail, whereas users only need to wait for less than one second if applying proposed PIH architecture and policies. In addition, the smaller the size IIMB occupies, the shorter average access time is. The reason is that miss penalty of PIB is greater than that of IIMB. The decrement of size of IIMB will increase the hit ratio of PIB and thus decrease the average access time. In Figure 6.22(b), we can see that access behaviors with sender and time locality has the shortest access time. Since the results of prior research that personal information access behaviors follows both time and sender locality, the proposed accessing policies can improve the performance of personal information management.

6.3 Summary



In this chapter, we have proposed a load-based scheduling scheme in a two-tier network architecture. The presented scheduling scheme makes use of three proxies to utilize the bandwidth of two-tier networks more effectively. Simulation results show that the presented scheme can help NEMOs in providing services with high completion rate and/or short average waiting time.

We also proposed the architecture and the accessing policies of PIH, and implemented a prototype of e-mail hierarchical system in this paper. The hierarchical architecture of PIH enables users to access and manage their personal information through mobile devices at any time in any place. The proposed specialized accessing policies consider all characteristics of different users, services, and devices. Different users can specify their personal preferences in the memory card. Different services have the proper abstraction strategies to reduce the size of packets to be transferred to or stored in the mobile devices. Different devices run different algorithms according to their storage space and processing power. The proposed architecture

and policies of PIH can help users flexibly manage their personal information on multiple devices and utilize the memory of all devices efficiently. In the future, we wish to implement a PIH system on a real mobile phone and study mechanisms to reduce the miss rate.



Chapter 7

Conclusion and Future Works

We have proposed mechanisms to solve network addressing issues in a hierarchical mobile network. To solve address auto-configuration problem, a DHCP relay method and a DHCP proxy method are proposed by configuring all hosts as DHCP relays or DHCP proxies. DHCP relay method can reduce signal overhead whereas DHCP proxy together with proposed PNAA can reduce both signal overhead and latency. DHCP proxy method extends the concept of the canonical factorization theorem to guarantee the uniqueness of addresses assigned by different nodes.

As for routing problems in hierarchical mobile network, we mainly adopted mobile IP to support seamless roaming. We made some modifications to support mobile hosts roam in private networks and support LFNs roam in hierarchical NEMOs. We also proposed two enhanced routing and scheduling mechanisms according to traffic load and type to further improve routing performance. The load-balanced routing protocol can balance traffic loads of all gateways and relieve the bottleneck problem. The load-based scheduling scheme schedules traffic according to service type and utilizes bandwidth of two-tier networks efficiently. In addition, a localization algorithm is also proposed, and location information of mobile nodes can help correctly route packets and reduce handoff latency.

In information access problem, we studied the successful experience of memory hierarchy in computer systems and proposed a Personal Information Hierarchy and corresponding accessing policies to help users manage their personal information ubiquitously. The proposed PIH and policies can utilize memory space efficiently and provide users both portability and storage capacity.

We believe the future trend will be the integration of various wireless networks. Although we have solved basic network addressing and access problems in hierarchical mobile networks, there should be other network management policies to be designed to provide reliable services to users. The concept of prime numbers can be extended to provide promising network management mechanisms to solve fault tolerant, quality of service routing, or self-healing problems in a multi-hop wireless network.



Bibliography

- [1] M. Audeh. Metropolitan-scale wi-fi mesh networks. *IEEE Computer*, December 2004.
- [2] A. N. C. E. Perkins, R. Wakikawa and A. J. Tuominen. Internet connectivity for mobile ad hoc networks. *Wireless Communications and Mobile Computing (WCMC)*, 2002.
- [3] H.-W. L. C.-F. Huang and Y.-C. Tseng. A two-tier heterogeneous mobile ad hoc network architecture and its load-balance routing problem. In *IEEE VTC*, 2003.
- [4] C. Perkins, E. Belding-Royer and S. Das. Ad hoc On-Demand Distance Vector (AODV) Routing. RFC 3561, July 2003.
- [5] C. Perkins, et al. IP mobility support for IPv4. RFC 3220, January 2002.
- [6] C. Perkins, et al. IP mobility support for IPv4. RFC 3344, August 2002.
- [7] J. Cai and K.-L. Tan. Energy-efficient selective cache invalidation. *ACM/Baltzer J. Wireless Networks (WINET)*, 5(6):489–502, 1999.
- [8] E. H. Callaway. *Wireless sensor networks: architectures and protocols*. Auerbach Publications, 2004.
- [9] C.E. Perkins, J.T. Malinen, R. Wakikawa, E.M. Belding-Royer and Y. Sun. IP Address Autoconfiguration for Ad Hoc Networks, draft-ietfmanet-autoconf-01.txt, IETF MANET Working Group, July 2000.
- [10] R. G. Chalermek Intanagonwiwat and D. Estrin. Directed diffusion: A scalable and robust communication paradigm for sensor networks. In *Proc. of the Sixth Annual International Conference on Mobile Computing and Networking*, pages 56–67, 2000.

- [11] D. Comer. *Internetworking with TCP/IP Volume I: Principle, Protocols, and Architecture Third Edition*. Prentice Hall International, 1995.
- [12] K. M. S. T. Z. C.S. Raghavendra. *Wireless sensor networks*. Kluwer Academic, 2004.
- [13] G. P. M. S. D. Estrin, L. Girod. Instrumenting the world with wireless sensor networks. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2001)*, 2001.
- [14] D. Johnson, C. Perkins, J. Arkko. Mobility Support in IPv6. RFC 3775, June 2004.
- [15] K. E. D. Ed. *Understanding GPS: principles and applications*. Artech House, 1996.
- [16] N. A. N. et al. Wireless convergence architecture: a case study using gsm and wireless lan. *ACM Mobile Networks and Applications*, August 2002.
- [17] D. Gries and F. B. Schneider. *A Logical Approach to Discrete Math*. Springer-Verlag New York, New York, NY, USA, 1994.
- [18] J. Gwizdka and TaskView. Design and evaluation of a task-based email interface. In *Proceedings of the 2002 Conference of the Centre for Advanced Studies on Collaborative Research*, page 4, 2002.
- [19] H. Levkowitz, S. Vaarala. Mobile IP Traversal of Network Address Translation (NAT) Devices, RFC 3519, April 2003.
- [20] H.-W. Cha, J.-S. Park and H.-J. Kim. Support of Internet Connectivity for AODV, draft-cha-manet-AODV-internet-00.txt, February 2004.
- [21] L. N. H. Zhou and M. Mutka. Prophet address allocation for large scale manets. In *Proc. INFOCOM*, 2003.
- [22] S. J. He T. Huang, C. Blum B.M. and A. T. Range-free localization schemes for large scale sensor networks. In *Proc. of MOBICOM*, 2003.

- [23] Y.-Y. Hsu and Y.-C. T. et al. Design and implementation of two-tier mobile ad hoc networks with seamless roaming and load-balancing routing capability. In *Proc. Quality of Service in Heterogeneous Wired/Wireless Networks (QShine'04)*, 2004.
- [24] Y.-Y. Hsu and C.-C. Tseng. A hierarchical proxy architecture with load-based scheduling scheme to support network mobility. In *Proc. of International Workshop on Applications of Ad Hoc Networks*, 2003.
- [25] Y. S. I. F. Akyildiz, W. Su and E. Cayirci. Wireless sensor networks: A survey. *Computer Networks*, 38(4):393–422, March 2002.
- [26] M. N. Inc. Mesh networks technology overview. <http://www.meshnetworks.com>.
- [27] H. J. and B. G. Location systems for ubiquitous computing. *IEEE Computer*, 34:57–66, Sep. 2001.
- [28] W. J. and L. C. Designing a positioning system for finding things and people indoors. *IEEE Spectrum*, 35:71–78, 1998.
- [29] D.-L. L. J. Xu, Q. Hu and W.-C. Lee. Saiu: An efficient cache replacement policy for wireless on-demand broadcasts. In *Proc. ACM CIKM*, page 46V53, 2000.
- [30] G. V. Laura Dabbish and J. Cadiz. Marked for deletion: an analysis of email data. In *Proceedings of Conference on Human Factors in Computing Systems, CHI '03 Extended Abstracts on Human Factors in Computer Systems*, 2003.
- [31] J. P. Macker and M. S. Corson. Mobile ad hoc networking and the IETF. *ACM Mobile Computing and Communications Reviews*, 4(4):12–13, Oct. 2000.
- [32] M. Mohsin and R. Prakash. Ip address assignment in mobile ad hoc networks. In *Proc. of IEEE MILCOM*, 2002.
- [33] C. S. R. Murthy and B. S. Manoj. *Ad Hoc Wireless Networks: Architecture and Protocols*. Prentice hall, 2004.

- [34] S. Nesargi and R. Prakash. Manetconf: Configuration of hosts in a mobile ad hoc network. In *Proc. INFOCOM*, 2002.
- [35] N. Networks. Nokia rooftop wireless routing - white paper. <http://www.americasnetwork.com/americasnetwork/data/articlebrief/americasnetwork/412002/34898/article.pdf>.
- [36] R. Networks. Reach for everyone: Meshworks. <http://www.radiantnetworks.com/meshworks/mwlesummary.asp>.
- [37] V. NH. Weak duplicate address detection in mobile ad hoc networks. In *Proc. ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'02)*, 2002.
- [38] B. P. and P. V.N. Radar: An in-building rf-based user location and tracking system. In *Proc. of INFOCOM*, 2000.
- [39] R. K. P. Ratanchandani. A hybrid approach to internet connectivity for mobile ad hoc networks. In *Proc. of IEEE Wireless Communications and Networking Conference (WCNC)*, 2003.
- [40] P. Thubert, M. Molteni. IPv6 reverse routing header and its application to mobile networks, draft-thubert-nemo-reverse-routing-header-00.txt, IETF NEMO Working Group, June 2002.
- [41] D. A. Patterson and J. L. Hennessy. *Computer Organization and Design Second Edition: The Hardware/Software Interface*. Morgan Kaufmann, 1997.
- [42] C. E. Perkins. *Ad Hoc Networking*. Addison-Wesley, Boston, MA, USA, 2001.
- [43] C. E. Perkins and P. Bhagwat. Highly dynamic destination sequenced distance vector (dsv) routing for mobile computers. In *Proc. of the SIGCOMM 94 Conference on Communications Architectures, Protocols and Applications*, 1994.
- [44] G. Pottie and W. Kaiser. Wireless sensor networks. *Communications of the ACM*, 43(5):51–58, May 2000.

- [45] R. Droms. Dynamic Host Configuration Protocol. RFC 2131, March 1997.
- [46] R. Ogier, F. Templin and M. Lewis. Topology Dissemination Based on Reverse-Path Forwarding (TBRPF). RFC 3684, February 2004.
- [47] T. S. Rappaport. *Wireless Communications, Principles and Practice, Second Edition*. Prentice Hall, 2002.
- [48] Q. Ren and M. H. Dunham. Using clustering for effective management of a semantic cache in mobile computing. In *Proc. of the International Workshop on Data Engineering for Wireless and Mobile Access (MobiDE'99)*, pages 94–101, 1999.
- [49] E. M. Royer and C.-K. Toh. A review of current routing protocols for ad hoc mobile wireless networks. *IEEE Personal Communications*, pages 46–55, Apr. 1999.
- [50] S. Corson and J. Macker. Mobile Ad hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations. RFC 2501, January 1999.
- [51] Y.-S. C. S.-Y. Ni, Y.-C. Tseng and J.-P. Sheu. The broadcast storm problem in a mobile ad hoc network. In *Proc. ACM/IEEE International conference on Mobile Computing and Networking*, 1999.
- [52] Q. J. Steve Whittaker and L. Terveen. Managing communications: Contact management: identifying contacts to support long-term communication. In *Proceedings of the 2002 ACM conference on Computer supported cooperative work*, 2002.
- [53] Y. Sun and E. M. Belding-Royer. A study of dynamic addressing techniques in mobile ad hoc networks. *Wireless Communications and Mobile Computing*, Apr. 2004.
- [54] T. Clausen, Ed. and P. Jacquet, Ed. Optimized Link State Routing Protocol (OLSR). RFC 3626, October 2003.
- [55] Thierry Ernst et al. Mobile networks support in mobile Ipv6 (Prefix Scope Binding Updates), draft-ernst-mobileip-v6-network-03.txt, IETF NEMO Working Group, March 2002.

- [56] Thierry Ernst, Hong-Yon Lach. Network mobility support requirements, draft-ernst-monet-requirements-00.txt, IETF NEMO Working Group, February 2002.
- [57] Thierry Ernst, Hong-Yon Lach. Network mobility support terminology, draft-ernst-monet-terminology-01.txt, IETF NEMO Working Group, July 2002.
- [58] C. E. L. Thomas H. Cormen and R. L. Rivest. *Introduction to Algorithms*. McGraw-Hill Book Company, 1999.
- [59] Timothy J. Kniveton et al. Mobile router support with mobile IP, draft-kniveton-mobrtr-02.txt, IETF NEMO Working Group, July 2002.
- [60] Timothy J. Kniveton, et al. Problem scope and requirements for mobile networks working group, draft-kniveton-monet-requirements-00.txt, IETF NEMO Working Group, February 2002.
- [61] Y.-C. Tseng and T.-Y. Hsieh. Fully power-aware and location-aware protocols for wireless multi-hop ad hoc networks. In *Proc. of the Int'l Conf. on Computer Communication and Networks (ICCCN)*, 2002.
- [62] A. N. L. X. and P. K. Performance of toa estimation algorithms in different indoor multipath conditions. In *Proc. of IEEE Wireless Communications and Networking Conference (WCNC)*, 2004.
- [63] C.-C. S. Y.-C. Tseng and W.-T. Chen. Mobile ip and ad hoc networks: An integration and implementation experience. *IEEE Computer*, 36(5), May 2003.
- [64] M.-L. T. Yuan-Ying Hsu and C.-C. Tseng. An adaptation from memory hierarchy to personal information management on multiple devices. In *Proc. of Mobile Computing Workshop*, 2004.
- [65] P. C. Z.-G. Zhou, R. Chen and A. Seneviratne. A software based indoor relative location management system. In *Proc. of Wireless and Optical Communications*, 2002.
- [66] F. Zhao and L. Guibas. *Wireless sensor networks: an Information Processing Approach*. Elsevier, 2004.