



2
2004
59713411

This is to certify that the
dissertation entitled

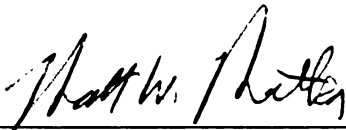
A Cooperative Ad Hoc Network to Support
Efficient Access to Internet Data

presented by

Seung-Seok Kang

has been accepted towards fulfillment
of the requirements for the

Doctoral degree in Computer Science



Major Professor's Signature

August 4, 2004

Date

LIBRARY
Michigan State
University

PLACE IN RETURN BOX to remove this checkout from your record.
TO AVOID FINES return on or before date due.
MAY BE RECALLED with earlier due date if requested.

| DATE DUE | DATE DUE | DATE DUE |
|----------|----------|----------|
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |

A COOPERATIVE AD HOC NETWORK TO SUPPORT
EFFICIENT ACCESS TO INTERNET DATA

By

Seung-Seok Kang

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

Department of Computer Science and Engineering

2004

ABSTRACT

A COOPERATIVE AD HOC NETWORK TO SUPPORT EFFICIENT ACCESS TO INTERNET DATA

By

Seung-Seok Kang

A CHUM (Cooperating ad Hoc networking to sUpport Messaging) network is an *ad hoc* network in which mobile devices cooperate to reduce connection costs when accessing wireless global networks. This dissertation presents how members of a CHUM network cooperate in order to download and to share data stored in the Internet, such as on-air streaming TV content, interactive mobile game programs, MP3 files, movie clips, etc. This dissertation assumes that mobile devices, called the peers, have two wireless interfaces, one for connecting to outside global networks and the other for forming an *ad hoc* network. When downloading multimedia content, one of the peers, called the proxy, downloads content via the WWAN connection and distributes the data using its WLAN channel. Each peer in a CHUM takes turns serving as the proxy. The multimedia CHUM network needs to maintain membership information and schedules the next proxy. In an one-tier CHUM network, there is no cooperation between the wireless network and the wired network. The proxy performs the maintenance task of the CHUM network. In contrast, a two-tier CHUM network

allows the cooperation between mobile peers and their associated servers in wired networks. The associated server of the proxy, called the master server, maintains peer membership information. With the support from the servers in wired networks, peers receive the benefits of saving telecommunication cost, *ad hoc* wireless bandwidth and computation power, which results in reducing the consumption of battery power. When downloading a file from the Internet, each participating peer may download a unique portion of the file via its WWAN connection, and exchange the portion with other peers in order to reconstruct the complete file. The servers in the wired network specify for each peer to download a specific portion of the file. The distribution of the portion should be carefully controlled, because all peers become senders and receivers at the same time. Uncontrolled distribution may cause a broadcast storm problem. Two distribution methods are introduced: per-packet based and per-peer based. In the per-packet based method, one packet from a peer is rebroadcasted over the *ad hoc* network at a time. In contrast, the per-peer based method allows each peer to take turns to transmit some of its packets only to its one-hop neighbor peers until all peers receive the complete content. In some special cases, a newly arrived peer may want to fetch some content already stored in an *ad hoc* CHUM network rather than from the Internet, which obviates repeated downloading of the content via the cost-based WWAN connection. In this case, the CHUM network becomes a content storage for the participating member peers. The CHUM network may also provide an anonymous connection between the source peer and the receiving peer in order to encourage privacy and to discourage traffic analysis. Two intermediate peers help establish the anonymous connection.

Copyright by
SEUNG-SEOK KANG
2004

To *my parents*

TABLE OF CONTENTS

| | |
|---|-----------|
| LIST OF FIGURES | ix |
| 1 Introduction | 1 |
| 2 Literature Review and Related Work | 9 |
| 2.1 Wireless local area network (WLAN) | 9 |
| 2.1.1 Contention-based WLAN | 10 |
| 2.2 <i>Ad hoc</i> networks | 17 |
| 2.2.1 <i>Ad hoc</i> routing protocol | 17 |
| 2.2.2 <i>Ad hoc</i> multicast protocol | 21 |
| 2.2.3 <i>Ad hoc</i> broadcast protocol | 24 |
| 2.3 Wireless wide area network (WWAN) | 27 |
| 2.3.1 1G networks | 28 |
| 2.3.2 2G networks | 29 |
| 2.3.3 2.5G networks | 30 |
| 2.3.4 3G networks | 31 |
| 2.3.5 4G/B3G networks | 33 |
| 2.4 Bluetooth | 34 |
| 2.5 CHUM related work | 36 |
| 3 One-Tier CHUM Network | 41 |
| 3.1 Introduction | 41 |
| 3.2 Network formation | 42 |
| 3.2.1 Neighborhood maintenance and data chumcasting | 45 |
| 3.3 Membership management | 47 |
| 3.4 Recovering from failures | 51 |
| 3.4.1 Recovery of a missing peer | 51 |
| 3.4.2 Proxy failure recovery | 52 |
| 3.4.3 Network partitioning recovery | 53 |

| | | |
|----------|--|-----------|
| 3.5 | State transitions | 55 |
| 3.6 | Summary | 57 |
| 4 | Two-Tier CHUM Network | 58 |
| 4.1 | Introduction | 58 |
| 4.2 | Network formation | 62 |
| 4.3 | $CHUM_2$ network management | 66 |
| 4.3.1 | Membership management | 66 |
| 4.3.2 | Selection of rebroadcasting peers | 70 |
| 4.3.3 | Proxy scheduling | 72 |
| 4.3.4 | Network partition management | 74 |
| 4.4 | Security | 76 |
| 4.5 | Buffer management in peers | 78 |
| 4.6 | Cost savings in a $CHUM_2$ network | 82 |
| 4.7 | Summary | 86 |
| 5 | Parallel Downloading of CHUM Networks | 87 |
| 5.1 | Introduction | 87 |
| 5.2 | Network operation | 89 |
| 5.2.1 | Formation of a $CHUM_p$ network | 89 |
| 5.2.2 | Membership management | 92 |
| 5.2.3 | Download scheduling | 93 |
| 5.3 | Per-packet based content distribution | 96 |
| 5.3.1 | Content distribution scheduling | 96 |
| 5.3.2 | Security issue | 103 |
| 5.3.3 | Simulation result of per-packet method | 106 |
| 5.4 | Per-peer based content distribution | 109 |
| 5.4.1 | Simulation results comparison | 117 |
| 5.5 | Summary | 126 |

| | |
|---|----------------|
| 6 Anonymous Sharing in CHUM networks | 127 |
| 6.1 Introduction | 127 |
| 6.2 $CHUM_a$ network operation | 131 |
| 6.2.1 File naming and mapping | 131 |
| 6.2.2 File distribution mechanism | 133 |
| 6.3 Simulation results | 141 |
| 6.4 Summary | 149 |
| 7 Conclusion and Future Work | 150 |
| 7.1 Conclusion | 150 |
| 7.2 Future Work | 152 |
| 7.2.1 Buffering in Peers | 152 |
| 7.2.2 QoS Enhanced Multimedia Downloading | 152 |
| 7.2.3 Parallel Downloading with Parallel Distribution | 153 |
| 7.2.4 Implementation of the CHUM network | 153 |
| BIBLIOGRAPHY | 155 |

LIST OF FIGURES

| | | |
|-----|--|----|
| 1.1 | Mobile devices accessing Internet through individual ISP connections. Each mobile device incurs telecommunication connection charges. . . | 2 |
| 1.2 | A single peer incurs telecommunication charges. The connection is shared by all peers. | 3 |
| 1.3 | The proxy migrates to another peer | 4 |
| 1.4 | Mobile devices download same target file in the Internet | 5 |
| 1.5 | Each peer downloads an assigned portion from the Internet | 6 |
| 1.6 | Downloaded portions are exchanged over the <i>ad hoc</i> connection | 7 |
| 2.1 | CSMA problems in WLAN | 11 |
| 3.1 | JOIN, ACCEPT, and HELLO Packet Format | 44 |
| 3.2 | REFRESH, LEAVE, GROUP packet format | 48 |
| 3.3 | Packet Delivery and Route Discovery | 49 |
| 3.4 | SCHEDULE and REBUILD Packet Format | 52 |
| 3.5 | State Transition of a $CHUM_1$ network | 56 |
| 3.6 | Packets Exchanged in a $CHUM_1$ Network | 57 |
| 4.1 | A single peer incurs telecommunication charges. The connection is shared by all peers. | 59 |
| 4.2 | The proxy migrates to another peer | 60 |
| 4.3 | JOIN, ACCEPT, and HELLO Packet Format | 64 |
| 4.4 | The sequence of introducing a mobile device to a $CHUM_2$ network . . . | 66 |
| 4.5 | The sequence of processing neighbor information in a $CHUM_2$ network . | 69 |
| 4.6 | The sequence of proxy role transition in a $CHUM_2$ network | 74 |
| 4.7 | A tree of data delivery from a proxy to all peers with some rebroadcasting peers | 80 |
| 4.8 | Cost of Each Peer Playing 30 min Streaming Data | 83 |
| 4.9 | Cost Savings of Each Peer with 20 Participating Peers | 84 |

| | | |
|------|---|-----|
| 4.10 | Average Cost Savings of Peers with Varying Number of Peers | 85 |
| 5.1 | The sequence of processing neighbor information | 92 |
| 5.2 | The sequence of generating rebroadcast peer set | 98 |
| 5.3 | A case of an ad hoc network topology | 99 |
| 5.4 | A triggering example of Data and DONE packet transmission | 101 |
| 5.5 | $CHUM_p$ network encryption and authentication | 104 |
| 5.6 | The total number of 3G payable packets with different number of peers . | 107 |
| 5.7 | Exchange completion time with different number of peers | 108 |
| 5.8 | Exchange completion time with different simulation area size | 109 |
| 5.9 | The number of 3G payable packets with different simulation area size . . | 110 |
| 5.10 | A 3-hop network topology and its transmission sequence circular list . . . | 112 |
| 5.11 | Exchange completion time with different number of packets sent at a time per peer | 114 |
| 5.12 | Exchange completion time with different network size. Larger transmit- at-a-time value decreases the completion time | 115 |
| 5.13 | Exchange completion time by increasing the number of hops | 119 |
| 5.14 | Exchange completion time by increasing the number of peers | 120 |
| 5.15 | The number of packets on the 3G link by varying the number of peers . . | 122 |
| 5.16 | Exchange completion time by varying the number of peers | 123 |
| 5.17 | The number of packets on the 3G link by varying the size of area | 124 |
| 5.18 | Exchange completion time by varying the size of area | 125 |
| 6.1 | Two peers transmit their partial content to requesting peer | 128 |
| 6.2 | The requesting peer downloads remaining content from $CHUM_a$ proxy . | 130 |
| 6.3 | Snapshot of FID table representing Fig. 6.2 | 134 |
| 6.4 | Overall sequence of anonymous connection setup | 136 |
| 6.5 | CIDS example of Fig 6.4 | 137 |
| 6.6 | Overall procedure of content verification and registration | 141 |
| 6.7 | Five simple topologies with several routes from peer 0 to peer 3 | 143 |
| 6.8 | Transmission completion time on five different cases | 144 |
| 6.9 | Completion time by varying the number of peers | 145 |
| 6.10 | Completion time of three different transmission method | 146 |
| 6.11 | Completion time vs. simulation area when $N=12$ and $N=16$ | 147 |
| 6.12 | Completion time vs. simulation area when packet size is 512B and 1024B | 148 |

Chapter 1

Introduction

Third generation (3G) telecommunication services promise to provide Internet access, such as downloading multimedia content, to mobile users having PDAs or mobile phones. These networks promise to provide service for both voice and data with the transmission speed of up to 2.05 Mbps [1] for a fee based on the amount of data traffic. Mobile users may download several types of data, such as multimedia streaming data, mobile game programs, and MP3 files. For example, in crowded areas such as airports, subway trains, stadiums, or bumper-to-bumper traffic, people may download multimedia data from the Internet, such as on-air TV shows, sports, and traffic information. Figure 1.1 illustrates the case where many mobile devices connect to the Internet via their telecommunication provider's 3G network, and then access their favorite services. Some popular multimedia data may be downloaded at the same time by several different people located within the same vicinity. Since it is expected that the cost to download multimedia data is a function of the amount of

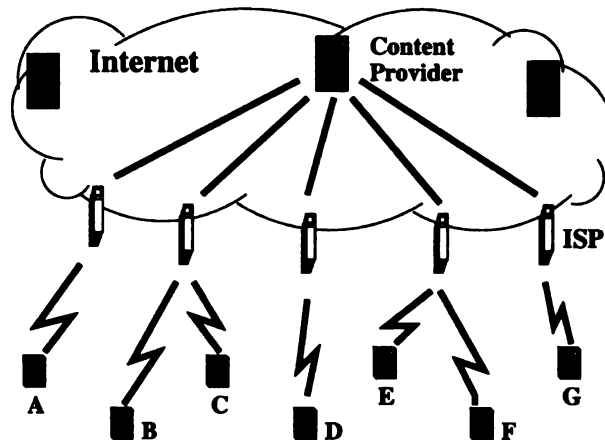


Figure 1.1: Mobile devices accessing Internet through individual ISP connections. Each mobile device incurs telecommunication connection charges.

data downloaded, then the costs of the telecommunication connections to access the Internet may be reduced by sharing the Internet connections among users located in a near vicinity. One possible scheme to share multimedia data is for mobile devices to form an *ad hoc* network and one of the mobile devices, called the *proxy*, connects to the Internet in order to download data and broadcasts it to the nearby devices in the *ad hoc* network. This may be possible if each mobile device has a 3G network interface for wireless wide area network (WWAN) access and a wireless local area network interface (WLAN), such as 802.11 or Bluetooth, to form an *ad hoc* network with users in the nearby region. Wireless LANs such as 802.11 [2] provide data rates of 11Mbps for 802.11b and 54Mbps for 802.11a and 802.11g, without a fee for data traffic. Bluetooth [3] is an open standard for two way radio communications between different devices. It provides connections between PDAs, notebooks, and mobile phones at the link speed of 1 Mbps.

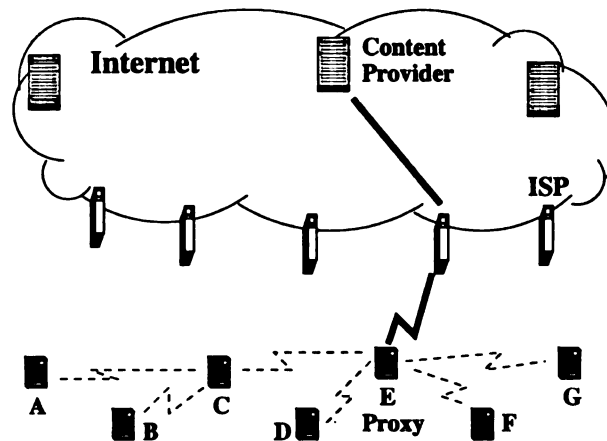


Figure 1.2: A single peer incurs telecommunication charges. The connection is shared by all peers.

To illustrate the operation of the *ad hoc* network, Figure 1.2 shows that the devices downloading the same content construct an *ad hoc* network and one of them, say E, becomes a proxy. As the proxy E receives data from its WWAN connection, it broadcasts the data to its neighbor peers with its WLAN. Some participating peers, for example C, may rebroadcast the packets to other peers, such as A and B, that are out of transmission range of the proxy. The proxy serves for a given amount of time and then schedules the next peer to serve as the proxy. This scheme aims to assure that only the peers that take turns serving as a proxy will be able to receive multimedia data. Figure 1.3 shows that peer B becomes the next proxy and peers D and F forward the received data packets for other peers. By sharing the connection and rebroadcasting downloaded data to others, the network reduces the connection costs for all participating peers.

The users of the mobile devices may form a similar network for different types of data such as programs, music files, and movie clips. Fig. 1.4 displays a situation

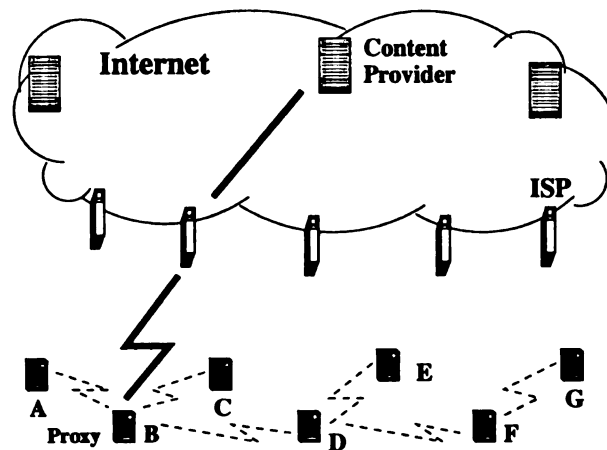


Figure 1.3: The proxy migrates to another peer

to download a file stored in the Internet. While the content provider (CP) in Fig. 1.1 serves multimedia streaming data, the CP in Fig. 1.4 stores a file to be downloaded to all mobile devices. In this case, the devices that download the same target file may form an *ad hoc* network in order to reduce the communication cost. The cost reduction scheme may be possible when each mobile device is assigned to download a given portion of the target file and shares its portion with other devices. The mobile devices may download their assigned portions of the target file via their 3G connections in parallel, and then construct an *ad hoc* network (that has no fee for data transferred) to exchange the remaining portions of the file.

Fig. 1.5 illustrates the mechanism for partitioning the target file and downloading the assigned portions by the mobile devices. Suppose, users of the mobile devices wish to download a mobile game program in order to play interactively with each other. Each mobile device is assigned a portion of the target file to download. Each device connects to its favorite ISP with its 3G link and contacts the CP with the

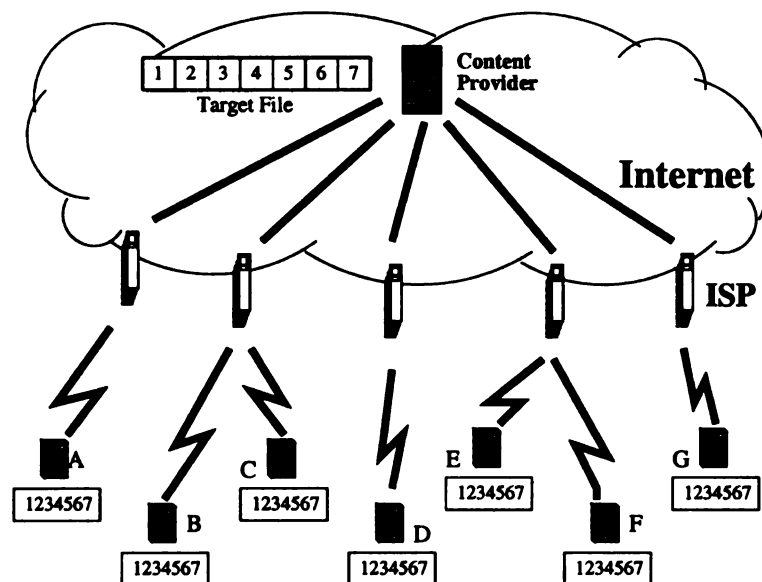


Figure 1.4: Mobile devices download same target file in the Internet

aid of its associated server in order to request its assigned portion of the file. The main difference between Fig. 1.4 and Fig. 1.5 is the thickness of the 3G connection, which represents the amount of data downloaded for each mobile device. In Fig. 1.4, each mobile device downloads the entire content from the CP. However, the thin line in Fig. 1.5 indicates that each device downloads only a portion of the target file. Upon completion of downloading the specified portion of the file, the mobile devices use their *ad hoc* connections to exchange their content with other member devices. Fig. 1.6 shows that all mobile devices participate by exchanging their partial content with others in order to reconstruct the target file.

The concept for sharing telecommunication links is called CHUM (Cooperating ad Hoc networking to sUpport Messaging). CHUM has previously been described as a way to support instant messaging [4]. It exploits ideas that are often described as

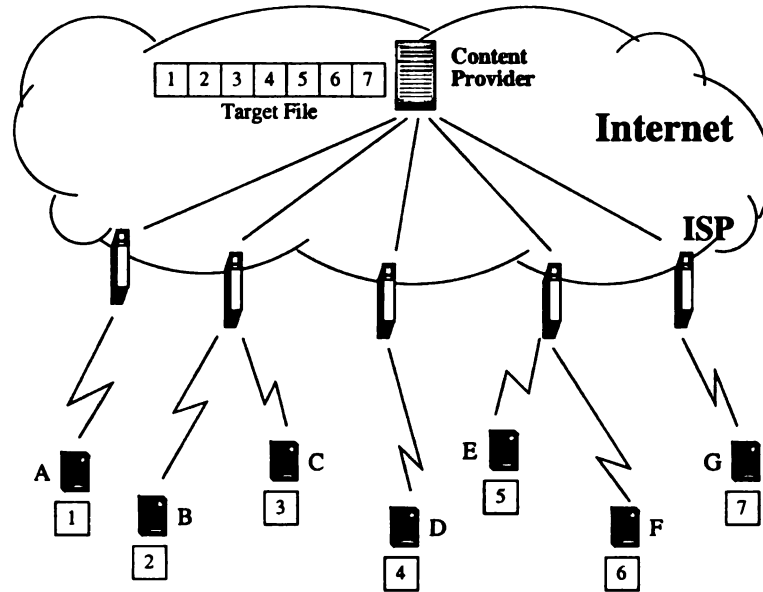


Figure 1.5: Each peer downloads an assigned portion from the Internet

peer-to-peer computing [5, 6]. There are several challenges to create and to manage the CHUM network. First, the cooperating mobile devices form a CHUM network to share the downloaded content. Second, the CHUM network requires a scheduling algorithm that decides which peers to download the Internet data. Moreover, the algorithm should adapt when new peers join the network. The third challenge is the distribution method of the downloaded data to all participating peers over the *ad hoc* network. The fourth challenge is to recover from network failure, such as peer failure and peer disappearance.

A CHUM network makes a few assumptions of the devices and the networking services available to the devices. First, the peers have networking interfaces that enable the formation of an *ad hoc* network even if there is not a wide-area Internet connection available at a given moment. No *ad hoc* routing algorithm nor unique IP

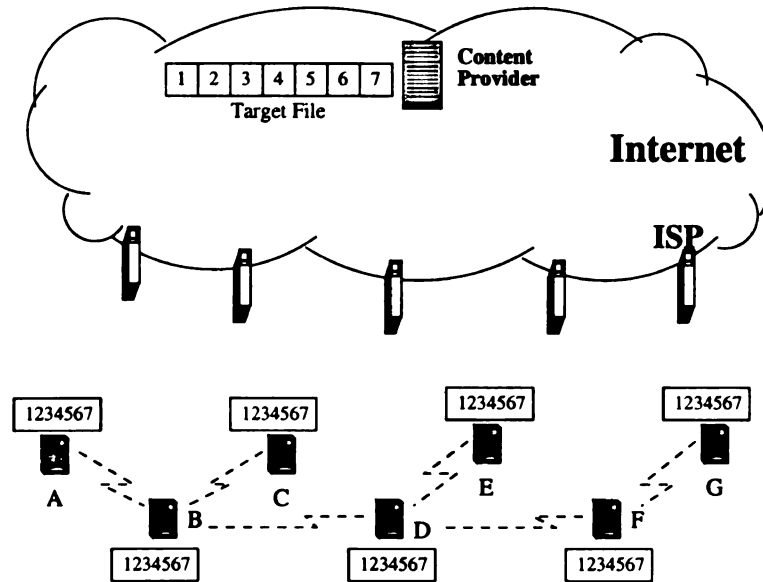


Figure 1.6: Downloaded portions are exchanged over the *ad hoc* connection

is needed, but each peer has a unique ID within the network. Note that almost all 802.11 interfaces support an *ad hoc* mode [2]. Bluetooth-enabled devices can form a piconet and/or scatternet for local area communication [3]. Second, the devices are equipped with an interface that enables a telecommunication connection to an ISP. While it is true that current handheld devices do not have multiple types of network interfaces available simultaneously, it is expected that this service will be available soon. Some PDA phones may have expansion packs for communication such as 802.11 and developments are underway to have dual Bluetooth and 3G cellular network options [7]. Third, users may join and leave the group at any time. Users who cooperate do not need to know each other. They may not even know of the presence of others who are cooperating. Fourth, all mobile devices share the same

wireless internal communication channel. They have omni-directional antennas and the same transmission range.

The rest of the dissertation is as follows. Chapter 2 describes related work. One-tier CHUM network is explained in chapter 3 and chapter 4 describes a two-tier CHUM network. Chapter 5 proposes a scheme of parallel downloading of partial content and of distributing the content. Chapter 6 presents an anonymous data transmission between peers in an *ad hoc* CHUM network. Finally, conclusion and future work are in chapter 7.

Chapter 2

Literature Review and Related Work

2.1 Wireless local area network (WLAN)

A wireless Local Area Network (WLAN) is a local area network system in which portable devices communicate through a wireless medium. The mobile devices may communicate either with a base station that is connected with a wired infrastructure, or without any installed infrastructures. When mobile devices form a network and exchange packets without a fixed infrastructure, it is called an *ad hoc* network. The base station is usually connected to wired networks such as the Internet and placed at the center of a cell. Each mobile device communicates directly with the base station in order to access wired networks or to exchange packets with other mobile devices.

The wireless medium is a limited resource and shared by all participating mobile devices in WLAN. The base station may coordinate to direct which device to use the medium at a specific time slot. This type of medium access control is contention-free, and FDMA (Frequency Division Multiple Access) and TDMA (Time Division Multiple Access) are some of the examples. Contention-free mechanism allows devices to use the wireless medium with a given frequency or at a given time slot(s). This mechanism provides efficient allocation of wireless resources to mobile devices. However, it also requires complex centralized control mechanism that is usually installed in the base station. Some WLANs including wireless ATM networks (WATM) [8, 9] adopts the contention-free medium access control (MAC) protocol.

Another wireless multiple access mechanism is a contention-based control protocol that includes CSMA, MACA, MACAW and 802.11. The contention-based MAC protocols do not need any centralized control entity in order to allocate wireless resources, but each mobile device competes to acquire the idle wireless medium. Each device has relatively simple contention logic to share the wireless channel. The contention-based MAC protocols can be applied to both cell-oriented base station networks as well as to *ad hoc* networks.

2.1.1 Contention-based WLAN

CSMA

One commonly used approach for contention-based MAC protocols is the Carrier Sense Multiple Access (CSMA) [10]. Current wired LANs use CSMA/CD, which

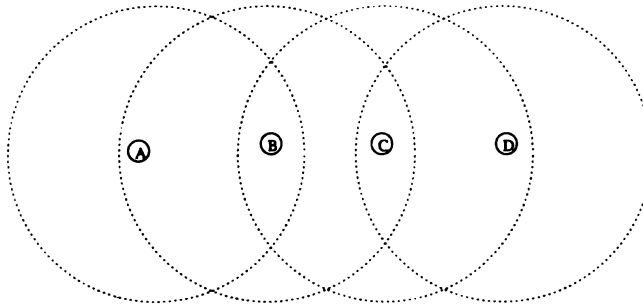


Figure 2.1: CSMA problems in WLAN

stands for Collision Detection. When a host detects that the transmission medium is idle, it sends a packet onto the medium and determines whether a collision has occurred. However, the traditional CSMA mechanism used in a wired LAN can not be applied to the wireless LAN due to the following problems.

- **Hidden Terminal Problem** [11]. Each host in a CSMA network senses the wireless medium before transmitting its packet. When the host detects a carrier, it defers its transmission and retries after some random time. The problem is, this protocol is not suitable for a wireless LAN because collisions occur at the receiving host and not at the sender. Two or more signals may collide at the receiver and carrier sense does not provide proper information for collision avoidance.

One of the problems of CSMA in WLAN is displayed in Figure 2.1. The circles denote the transmission range of each mobile host. Suppose host A is transmitting a packet to host B. Host B can hear host A, but host C can not hear host A's signal. That is, while host A sends a packet to B, host C may send a packet to host B because C does not sense any carrier from host A. Because host C is too far (hidden) from

host A, host C's carrier sense did not provide the necessary information. This is a typical scenario of the "hidden terminal" problem.

- **Exposed Terminal Problem.** There is another medium utilization problem in WLAN with CSMA. Suppose host B is transmitting a packet to host A in Figure 2.1. When host C tries to send a packet to host D and senses a carrier, the host may defer its transmission in CSMA. However, host C has no reason to defer its transmission to host D because host A is out of range of host C. Because host C is exposed to host B, host C falsely defers its transmission, which is called the "exposed terminal" problem [12].

MACA and MACAW

CSMA allows mobile hosts to provide information about possible collisions at the sending host but not at the receiver. The collision detection mechanism, which is used in wired LANs, can not be used in wireless network. Karn [13] proposed an approach, named MACA, to solve the problems of hidden terminal and exposed terminal. Multiple Access Collision Avoidance (MACA) was used as a basis for the IEEE 802.11 WLAN standard.

MACA uses two small-sized packets called Request-To-Send (RTS) packet and Clear-To-Send (CTS) packet. When a sender has a packet to transmit, it first sends a RTS packet to its neighbors. The RTS packet includes the size of the data packet that will follow as well as the destination address. The destination host replies with a CTS packet that contains the copied data packet size.

In Figure 2.1, suppose host A wants to send a packet to host B. Then, host A sends a RTS packet to its neighbors. If any host except host B receives the RTS packet, it must wait long enough for the CTS packet to be replied with no collision. As a response, host B replies with a CTS packet to host A. Any host hearing the CTS packet must remain silent during the following data transmission. In MACA, the hidden terminal problem is solved because host C will wait after receiving the CTS packet until the termination of host A's transmission. The exposed terminal problem is also solved because host C will hear only the RTS packet, and resume its transmission after the reception of host A's CTS packet. Unless host C sends a packet to host B, it is free to send to other hosts including host D after host B's reception of CTS packet.

Bharghavan, et al. in [14] proposed improvements on MACA and the enhanced media access protocol was called MACAW. MACAW includes the execution of the backoff algorithm individually for each stream, rather than for each host to improve the fairness. In addition, they suggests Multiplicative Increase and Linear Decrease (MILD) of the backoff algorithm upon a collision. MACAW also includes the execution of the backoff algorithm individually for each stream, rather than for each host for improving the fairness. In MACA, when a packet is lost, the sender detects the lost by the notification from the receiver's transport layer, which takes longer than from the MAC or data link layer. MACAW introduces an ACK packet that is transmitted after the successful data packet reception. Moreover, MACAW adds the

functionality of copying the congestion level from overheard packets in order to make the backoff algorithm behaves less violently and to improve network performance.

IEEE 802.11

IEEE 802.11 [15, 16] is a standard for WLAN and is limited in scope to the physical layer (PHY) and the medium access control (MAC) sublayer that origins to IEEE 802.3 Ethernet standard. The physical layer is the interface between the MAC and the wireless medium and defines radio modulation techniques, such as Frequency Hopping Spread Spectrum (FHSS), Direct Sequence Spread Spectrum (DSSS) and Orthogonal Frequency Division Multiplexing (OFDM), and a infrared specification such as IrDA. The MAC sublayer provides a controlled access to the shared wireless medium through two different access mechanism: the Distributed Coordination Function (DCF) and the Point Coordination Function (PCF). The DCF is the basic access mechanism of the IEEE 802.11 and it is mandatory for all hosts. The PCF is an optional extension to DCF and provides a time division duplexing capability to accommodate time bounded connection oriented service.

Hosts exchange RTS-CTS packets prior to a data packet transmission and ACK packet is returned by the receiving host when the data packet arrives intact. Transmissions before the RTS packet or after the ACK packet must wait at least one DCF Inter Frame Space (DIFS). The period between exchange of RTS-CTS, CTS-DATA, and DATA-ACK packet is one Short Inter Frame Space (SIFS) that is smaller than DIFS. The RTS packet reserves the wireless medium and the reservation

information is stored in the Network Allocation Vector (NAV) of all station detecting the packet.

A Basic Service Set (BSS) is a set of hosts that communicates with one another. When all hosts are mobile and there is no connection to a wired network, the BSS is called independent BSS (IBSS). Another name for IBSS is called the peer-to-peer mode or the *ad hoc* mode. When there is an access point (AP) in BSS, the BSS is called infrastructure BSS. An Extended Service Set (ESS) is a set of infrastructure BSSs, where the APs communicate among themselves to forward packets from one BSS to another and to facilitate the movement of mobile hosts from one BSS to another.

After IEEE 802.11 was standardized in July 1997, more standards for enhanced data rate are included in IEEE 802.11 family such as 802.11a and 802.11b. Both enhanced standards are approved in September 1999, and more supplementary to PHY and MAC standards are being developed.

- **IEEE 802.11** [15]. This is a standard for both PHY and MAC layer in 2.4 GHz ISM (Industrial Science Medical) band. It defines three PHY layer specifications such as FHSS, DSSS, and Infrared and two MAC layer specifications including DCF and PCF. This standard defines available bandwidth of 83.5MHz and provides the data rate of 1 Mbps and 2 Mbps. 802.11 provides encryption mechanism that is known as Wired Equivalent Privacy (WEP). For data encryption, the standard provides for optional encryption using a 40-bit shared-key RC4 PRNG algorithm from RSA Data Security.

- **IEEE 802.11b** [2]. This is a PHY layer standards that extends the IEEE 802.11 DSSS, providing higher data rates of 5.5 Mbps and 11 Mbps through the use of a more complex modulation technique. To accomplish this data rate, DSSS had to be selected as the PHY layer since FHSS cannot support the higher speed without violating current FCC regulation, and interoperates only with 1 Mbps and 2 Mbps 802.11 DSSS system. Ideally, users connect at the full 11 Mbps rate. However, when devices move beyond the optimal range for 11 Mbps operation, or if substantial interference is present, 802.11b hosts will transmit with lower speeds, falling back to 5.5, 2, and 1 Mbps.

- **IEEE 802.11a** [17]. IEEE 802.11a defines requirements for a PHY layer operating in the 5.0 GHz U-NII (Unlicensed National Information Infrastructure) band and data rates ranging from 6 Mbps to 54 Mbps [18]. As the distance between an AP and a mobile host increases, the data link rate of 802.11a drops quickly. However, 802.11a has similar range compared to 802.11b up to around 200 feet (60 meters). In addition, for the distances up to 200 feet in a typical office environment, 802.11a are 2 to 5 times better than 802.11b. The PHY layer of this standard adopts OFDM system that gives high performance and robustness against the adverse effects of multipath propagation. IEEE 802.11a has 8 indoor channels and IEEE 802.11b has 3 according to FCC regulations, which allows 802.11a network provides 8 times the throughput on 802.11b. This increase results from the fact that there was no co-channel interference in the 802.11a due to the availability of 8 channels in 8-cell system. Because of the different frequency usages, 802.11a and 802.11b can coexist

and interoperable in the same space. However, in order to achieve its maximum link layer speed, 802.11a needs more APs than that of 802.11b.

- **IEEE 802.11g.** IEEE 802.11g specifies a high-speed wireless LAN for data rates of up to 54 Mbps. It is backward compatible with 802.11b, which uses Complementary Code Keying (CCK) for 11 Mbps. For higher data rates, OFDM and as an option Packet Binary Convolutional Coding (PBCC) are employed [19]. IEEE 802.11e is working on an enhanced MAC with QoS provisions, and 802.11i on enhanced security functionality.

2.2 *Ad hoc* networks

An *ad hoc* network is a rapidly deployable wireless network in which there is no centralized control mechanism such as fixed routers and routing backbones. Usually, the hosts in *ad hoc* networks are mobile and the underlying communication medium is wireless. Each mobile host may work as a router. Such *ad hoc* networks may arise in personal area networking, outdoor events, conference rooms, rescue operations, battlefield operation etc.

2.2.1 *Ad hoc* routing protocol

Traditional wired routing protocols such as link state routing [20] and distance vector routing protocol [21, 22] are not suitable for *ad hoc* networks. An *ad hoc* network is a dynamically reconfigurable wireless network without fixed infrastructure and no

explicit lines installed between mobile hosts. Each mobile host uses a broadcast mechanism rather than sending packets to each outgoing link in wired network. If an *ad hoc* network adopts one of the wired network routing protocols that periodically needs to exchange neighborhood information, each host may be regarded as a neighbor of many other hosts and many redundant neighborhood information may be exchanged. The periodic exchange of packets may consume more battery power, which is a limited critical resource in a mobile device. Moreover, the movement of mobile hosts changes the network topology more often than that of wired networks. In addition, a wired receiver host can reply back to the sender host on the same link, but a wireless host may not reply if its transmission range is different from the sender's range. In *ad hoc* networks, if a receiving host is out of range of a sender, some intermediate hosts help forwarding the sender's packets. Since the advent of DARPA packet radio networks in the early 1970s [23], several *ad hoc* routing algorithms are introduced and studied. Some of the protocols use tables, such as Destination-Sequenced Distance Vector routing protocol (DSDV) [24], Cluster-head Gateway Switch Routing (CGSR) [25], Wireless Routing Protocol (WRP) [26], Fisheye State Routing (FSR) [27] etc. The table-driven protocols keep maintaining up-to-date route information to every other host in the network. In contrast, on-demand routing protocols discover routes only when the source host needs, which includes Dynamic Source Routing (DSR) [28], Ad hoc On-demand Distance Vector routing protocol (AODV) [29], Temporally-Ordered Routing Algorithm (TORA) [30], Associativity-Based Routing (ABR) [31]. Some routing protocols use Location or Geographic information, such as Location-Aided Routing (LAR) [32], Greedy Perime-

ter Stateless Routing (GPSR) [33], Distance Routing Effect Algorithm for Mobility (DREAM) [34], Zone Routing Protocol (ZRP) [35, 36], etc.

- **Flooding.** Flooding [12] starts with a sending host transmitting a packet to all neighbors. Any host receiving the packet examines the destination address in the packet and rebroadcasts if the destination address is different from the host's. This forwarding events continue until all reachable network hosts have received the packet. Flooding is the simplest protocol to implement and it guarantees to deliver packets with the shortest path. However, flooding requires many redundant packets transmitted, which causes unnecessary wireless bandwidth consumption and packet collisions. Even though flooding is not suitable for an *ad hoc* routing protocol, this mechanism is necessary as a part of other routing and multicasting protocols such as a destination host discovery.

- **DSDV.** Destination-Sequenced Distance Vector (DSDV) routing protocol [24] borrows ideas from the distance vector routing algorithm in wired networks. Each host maintains all available destination hosts with the next host and the number of hops to each destination. The routing information is exchanged with neighbor hosts with a sequence number. When a host receives new routing information, it compares with the previous routing packet. The host prefers the information with a larger sequence number. If the two routes have the same sequence numbers, the route with small number of hop wins. When a host decides the route to a destination is broken, the host sets the number hops with an infinite value, then it propagates the informa-

tion to its neighbors after incrementing the sequence number by one. This increment causes other hosts to update their routing information of the broken route.

- **DSR.** Dynamic Source Routing (DSR) [28, 37] makes the source host to have complete route information and to place the information into each packet transmitted. When a host tries to send a packet to a destination host for which it does not know the route, the host performs a route discovery process. Route discovery starts by flooding the *ad hoc* network with the route request (RREQ) packet that contains a destination address. A host receiving RREQ packet rebroadcasts the packet attaching its address as a forwarding host if it does not have a path to the destination. Otherwise if a host receiving the RREQ packet is the destination host or it has a route information in its route cache, it responds with a route reply (RREP) packet including the path to the destination. The RREP packet routes back to the source host by traversing the RREQ packet backwards. The source host may maintain several routes to the destination and selects the most preferable one such as the path with the smallest number of hops. If any intermediate host in the source-routed path finds a broken link, it notifies by sending a route error (RERR) packet. The source deletes all route information related to the broken link, and attempts to retransmits with other routes, if exists, or invokes the route discovery process to find a new path.

- **AODV.** In *Ad hoc* On-demand Distance Vector routing protocol (AODV) [29, 38], a source host requests a destination route as needed like DSR, and each intermediate host maintains the next hop information to the destination. A source host in need of a route broadcasts a route request (RREQ) packet to its neighbors. Each

host records the address of the neighbor from which it received the RREQ packet, which sets up the reverse path to the source. Any host having a current route to the destination, including the destination host itself, unicasts a route reply (RREP) packet back to its neighbor through the reverse path of the RREQ packet. When an intermediate host receives a RREP packet, it sets the next hop to the destination with the neighbor host that transmits the RREP packet. Once the source host receives the RREP packet, it resumes sending packets along this route to the destination. When a host detects a link break in an active route by a timeout mechanism, it transmits an unsolicited RREP packet containing a list of all the destination which are now unreachable. Upon receiving the notification of the link breakage, source hosts reinitiates the route discovery process if they need a route to the destination.

2.2.2 *Ad hoc* multicast protocol

Multicasting is a packet delivery mechanism from a sending host to a group of receiving hosts. Multicast routing becomes one of the important networking features providing applications like multimedia contents distribution, teleconferencing, and distributed games. Several multicast protocols for wired networks are developed including Distance Vector Multicast Routing Protocol (DVMRP) [39], Core Based Tree (CBT) [40], Multicast Open Shortest Path First (MOPSF) [41], and Protocol Independent Multicast (PIM) [42]. They may not be properly applied to the *ad hoc* network for multicasting because frequent topology changes due to mobility of hosts may trigger frequent adjustments of the multicast tree structures. In addition, a

multicast tree needs a global routing topology of a network. The frequent exchange of routing information due to the network topology changes produces excessive consumption of wireless bandwidth and battery power. To overcome these limitations and to adapt to the mobile environment, several *ad hoc* multicast routing protocols are developed, which include On-Demand Multicast Routing Protocol (ODMRP) [43], Multicast Ad hoc On-Demand Distance Vector (MAODV) [44], Reservation-Based Multicast routing protocol (RBM) [45], Lightweight Adaptive Multicast algorithm (LAM) [46], Ad hoc Multicast Routing Protocol (AMRoute) [47], Ad hoc Multicast Routing protocol utilizing Increasing id-numberS (AMRIS) [48], Core Assisted Mesh Protocol (CAMP) [49, 50, 51], Adaptive Demand-driven Multicast Routing protocol (ADMR) [52], Forwarding Group Multicast Protocol (FGMP) [53], Bandwidth Efficient Multicast Routing Protocol [54], Multicast Core Extraction Distributed Ad Hoc Routing (MCEDAR) [55], Neighbor Supporting Ad hoc Multicast Routing Protocol (NSMRP) [56], and Dynamic Core Based Multicast Routing Protocol (DCMP) [57].

- **ODMRP**. On-Demand Multicast Routing Protocol (ODMRP) [43, 58] creates a mesh of forwarding group of hosts which forward multicast packets via flooding within the mesh. It uses a soft state approach in group maintenance. Members are refreshed as needed and do not send an explicit leave message. In ODMRP, group membership and multicast routes are established and updated by the source host on demand. When a multicast source has data to send, but do not have routing or membership information, they flood a **Join Data** packet. When a host receives a non-duplicate **Join Data**, it stores the upstream host ID and rebroadcasts the packet.

When the Join Data packet reaches a multicast receiver, the receiver creates a Join Table and broadcasts to the neighbors. When a host receives a Join Table, it checks if the next host ID of one of the entries matches its own ID. If it does, the host realizes that it is on the path to the source and thus is part of the forwarding group. It then broadcasts its own Join Table built upon matched entries. The Join Table is thus propagated by each forwarding group member until it reaches the multicast source via the shortest path. This process constructs the routes from a source to receivers and builds a mesh of hosts, the forwarding group. Multicast senders refresh the membership information and update the routes by sending Join Data periodically.

- **MAODV**. Multicast Ad hoc On-Demand Distance Vector (MAODV) [44] is an on-demand multicast routing protocol. However, unlike ODMRP, MAODV is tree based. Route discovery is based on a route request (RREQ) and route reply (RREP) cycle. When a multicast source requires a route to a multicast group, it broadcasts a RREQ packet with the join flag set and the destination address set to the multicast group address. A member of the multicast tree with a current route to the destination responds to the request with a RREP packet. Non members rebroadcast the RREQ packet. Each host upon receiving the RREQ updates its route table and records the sequence number and next hop information for the source host. This information is used to unicast the RREP back to the source. If the source host receives multiple replies for its route request, it chooses the route having the least hop count or the freshest sequence number. It then sends a multicast activation message (MACT). The MACT is used to activate the path from the source host to the host sending the

reply. If a source host does not receive a MACT message within a certain period, it broadcasts another RREQ. After a certain number of retries, the source assumes that there are no other members of the tree that can be reached and declares itself the group leader. The group leader has the function of periodically broadcasting group hello messages to maintain group connectivity. Hosts also periodically broadcast hello messages with a time-to-live value of one to maintain local connectivity.

2.2.3 *Ad hoc* broadcast protocol

Broadcasting is a process in which one host sends a packet to all other hosts in the network. Broadcasting is often necessary in mobile *ad hoc* network routing protocols. Many *ad hoc* unicast routing protocols use broadcasting to establish routes. Currently, most of the unicast and multicast protocols rely on a simplistic form of broadcasting called flooding in which each host retransmits each received unique packet exactly once. The main problems with flooding are that it typically causes unproductive and often harmful bandwidth congestion, as well as inefficient use of host resources such as computing cycles and battery power. Several broadcasting techniques are proposed whose goal is to minimize the number of retransmissions while attempting to ensure that a broadcast packet is delivered to each host in the network. The broadcast protocols use several techniques [59], such as simple flooding [60], probability and area based method [61], and neighbor knowledge method including pruning [62, 63], core extraction distributed *ad hoc* routing (CEDAR) [64], scalable broadcast algorithm (SBA) [65], multipoint relaying [66], and *ad hoc* broadcast protocol (AHBP) [67].

Probability based protocol

Probabilistic scheme. In this scheme, when a host receives a packet, it rebroadcasts the packet with probability P that is predetermined [61]. If $P=1$, the scheme is the same as flooding. In a dense network, having some hosts not rebroadcast due to the probability saves network resources without degrading packet delivery effectiveness over the network. In a sparse network, however, hosts may not receive all broadcast packets unless the value P is high in this scheme.

Counter-based scheme. When a host receives a packet, it tries to rebroadcast the packet but to be delayed due to a busy medium or the random backoff algorithm. During the delay period, the host may receive redundant packets from other rebroadcasting hosts before it resumes transmitting the packet. This scheme requires a host to initialize a counter value to one and to set a random period upon the reception of a new broadcast packet [61]. During the random period, the counter is increased by one for the same rebroadcast packet received. When the period expires, the host rebroadcasts the packet if the counter value is less than a threshold value. Otherwise, the host drops the packet. In a dense network, some hosts may drop the received packets. In a sparse network, all hosts may rebroadcast.

Neighbor knowledge based protocol

- **SBA.** The main idea of the Scalable Broadcast Algorithm (SBA) [65] is for a host to suppress a broadcast packet if all neighbors of the host have been received the packet by previous transmissions. SBA requires that all hosts need to know two-hop

neighbors. Two-hop neighbor information can be collected by exchanging periodic HELLO packets. The packet includes the address of the originating host and the list of its discovered neighbors. When a host receives HELLO packets from all its neighbors, it also knows its two-hop neighbors. When a host receives a broadcast packet, it may determine the one-hop neighbors that also have received the packet, and it adds the list of the neighbors covered by the previous transmission to the cover set. If the host has neighbors that do not receive the broadcast packet, it schedules the packet for transmission after a random delay. When the host receive the same packet from another host, it adds the list of neighbors covered by the transmission to the cover set. This process continues until either the delay expires and the packet is rebroadcasted, or the transmission is cancelled because all its neighbors are in the cover set.

- **MPR.** Multipoint Relaying (MPR) [66] is a deterministic method for reliable broadcasting. This method assumes that each host knows its two-hop neighbors. A host explicitly selects a minimal set of one-hop neighbors, called the multipoint relays (MPRs), and they are the only hosts that are allowed to rebroadcast packets from the host. Each node computes its own set of MPRs in a distributed manner in order to efficiently reach all hosts within the two-hop neighborhood, and the selection process is independent of other hosts' selection of MPRs. The computation of this minimal set is a NP-complete problem [68]. The authors in [66] proposed a heuristic greedy algorithm to find the minimal MPR set. In order to find a MPR set, the algorithm first finds any two-hop neighbors that can be accessed only by single one-hop neighbor, and add these one-hop neighbors to the MPR set. The covered two-hop neighbors

are placed in the cover set. Then, it selects one-hop neighbors, not in the MPR set, that would cover the most two-hop neighbors not in the cover set, until all two-hop neighbors are in the cover set.

- **AHBP**. Ad Hoc Broadcast Protocol (AHBP) [67] uses a term Broadcast Relay Gateway (BRG) that is similar to MPR. In AHBP, packets are rebroadcasted only by BRGs that constitute a connected dominating set (CDS) of the network. BRGs are computed on-demand and no virtual backbone structure needs to be maintained. BRGs are selected along with the propagation of broadcast packets. A source host announces the BRGs of one-hop neighbors packed in the header of each broadcast packet. If a host is listed in the BRGs, it utilizes two-hop neighbor knowledge to decide next hop BRGs by eliminating the covered host from the previous transmission. This allows a host to compute the most effective BRG set at the time a broadcast packet is transmitted.

2.3 Wireless wide area network (WWAN)

A Wireless Wide Area Network (WWAN) [1] is a fully bi-directional wireless network that covers cities, countries, and continents. WWANs are the primary means for nomadic users to access the data services. The first generation, or 1G, network comprised analog and voice based mobile systems that emerged in early 1980s. These systems were typically limited in capacity. The second generation (2G) appeared about 10 years later, with the first digital mobile, circuit-switched networks. The second generation system also supports simple nonvoice services like the Short Mes-

sage Service (SMS). To get from 2G networks to 3G services, wireless operators are adopting an intermediate step known as 2.5G system. This system gives subscribers an always-on, high-speed wireless access without requiring that service providers undergo the system-wide upgrades demanded by 3G. This intermediate step also relieves a burden for an industry from outrageous spectrum bidding wars. The third generation system will offer data rates high enough for mobile users to work with multimedia web content, video conferencing, and e-commerce [69]. While the standardization of 3G is still ongoing, the discussion of technical issues beyond 3G (B3G/4G) has already started [70, 71]. The major distinction of 4G over 3G communications is increased data transmission rates reaching approximately 100 Mbps. The fourth generation system is expected to deliver more advanced versions of the same improvements promised by 3G, such as enhanced multimedia, smooth streaming video, universal access, and portability across all types of devices.

2.3.1 1G networks

The introduction of cellular systems in the late 1970s and early 1980s represents a big jump in mobile communication, especially in capacity and mobility. Semiconductor technology and microprocessors made smaller, light weight, and more sophisticated mobile systems. The 1G cellular systems transmit only analog voice information. The 1G analog systems operate in the 800 MHz frequencies. The most prominent 1G systems are Advanced Mobile Phone System (AMPS) [72], Nordic Mobile Telephone (NMT), and Total Access Communication System (TACS) [73].

2.3.2 2G networks

The development of 2G cellular systems was driven by the need to improve transmission quality, system capacity, and coverage. Further advances in semiconductor technology and microwave devices brought digital transmission to mobile communications. The second generation systems operate in both the 800 MHz (806 to 902 MHz) and PCS bands (1850 to 1990 MHz). The second generation technology allows the communication that is digital, circuit-based, and narrowband but suitable for voice and limited data communications. Cellular systems use three different techniques for sharing a radio frequency (RF) spectrum such as FDMA, TDMA, and CDMA. TDMA and CDMA are the dominant techniques.

- **TDMA systems.** Time Division Multiple Access (TDMA) based systems divide an RF channel into time slots and allocates those slots to multiple calls. TDMA divides a 30 KHz channel into six time slots that are allocated in pairs, resulting in three simultaneously usable TDMA channels. Any given conversation can use one or more of every third time slot on an ongoing basis during a call. TDMA was defined by the IS-54 standard Digital AMPS (D-AMPS) in North America. GSM (Global System for Mobile communications) and the Japanese PDC (Personal Digital Cellular) are variants of TDMA.

- **CDMA systems.** Code Division Multiple Access (CDMA), developed by Qualcomm [74], is a technology that uses digital encoding and spread spectrum RF techniques to let multiple users share the same RF channel. CDMA divides the radio spectrum into channels that are 1.25 MHz wide. CDMA differentiates each user's

signal by encoding it uniquely. At the transmitting end, this is used to encode the signal, which is spread across the frequency spectrum. At the receiving end, code is detected and used to extract the user's information.

- **GSM system.** Global System for Mobile communications (GSM) became one of the popular 2G system due to its international roaming capability [73, 75]. Most GSM systems operate in the 900 MHz and 1.8 GHz bands, except in North America where they operate in the 1.9 GHz band. GSM uses a combination of FDMA and TDMA scheme on its 25-MHz-wide frequency spectrum. FDMA divides the 25 MHz bandwidth into 124 carrier frequencies of 200 KHz each. Each 200 KHz channel is then divided into eight time slots using TDMA rather than three slots. GSM systems support data speeds of 9.6 Kbps and the SMS feature.

2.3.3 2.5G networks

The technology used in 2.5G is developed as an upgrade to 2G network. The 2.5G system is packet-based and has more bandwidth than 2G as high as 384 Kbps. The systems can use many existing infrastructure, and thus can be implemented faster and less expensive than 3G.

- **GPRS system.** General Packet Radio Service (GPRS) [76, 77, 78] is one upgrade of GSM and introduces as a packet-switched system to transport high speed data efficiently over GSM- and TDMA-based networks. GPRS uses eight time slots in the 200 KHz channel and can support IP-based packet data speeds up to about 115 Kbps. However, if a single user would take over all eight time slots without any

error protection, the theoretical maximum GPRS data transmission speed of about 170 Kbps are achievable. GPRS reallocates several GSM time slots from voice to data uses, which leads to increasing data rates but decreasing voice rates. GPRS uses the GSM network to look up the location-register databases to obtain subscriber-profile data. Enabling GPRS on a GSM network will also require a new air interface for packet-switched traffic.

- **EDGE system.** Enhanced Data rates for GSM Evolution (EDGE) [79, 80] is another upgrade to GSM and is designed to provide data rates up to 473 Kbps using the same 200 KHz TDMA carrier. It uses 3G transmission technology but works in GSM's frequency range. In addition, EDGE works with new modulation scheme known as Eight Phase Shift Keying (8PSK). The 8PSK scheme provides for up to 3 bits per symbol, rather than GPRS's 1 bit per symbol, which facilitates an up to three times improvement over GPRS.

2.3.4 3G networks

The main characteristics of 3G systems, known collectively as IMT-2000, are a single family of compatible standards that promise to offer high data rates up to 2 Mbps [81]. Among the several proposals of IMT-2000 standards, the most important proposals are the UMTS (W-CDMA) [82, 83, 84] as the successor to GSM, CDMA2000 as the IS-95 successor, and TD-SCDMA as TDMA based enhancements to D-AMPS/GSM. UMTS and CDMA2000 are not compatible [85] from the perspective that they have different chip rates: UMTS for 3.84 Mcps and CDMA2000 for 1.28 Mcps. A newly

introduced UTRA mode, multicarrier (MC), is expected to establish compatibility between UMTS and CDMA2000.

- **UMTS (W-CDMA)**. Universal Mobile Telecommunications System (UMTS), also referred to as Wideband CDMA (W-CDMA) [86, 87, 88], is envisioned as the successor to GSM. The UMTS network incorporates an wireless link that uses a 5-MHz-wide carrier to enable to support transmission speed up to 2 Mbps per mobile user and establishes a global roaming standard.

- **CDMA2000**. CDMA2000 [89, 90] is a CDMA upgrade. Initially, CDMA2000 1x uses only a 1.25 MHz channel, but with CDMA2000 3x, three 1.25 MHz channels can be combined to form a super channel structure, which would provide the 3G maximum throughput of 2.05 Mbps. This system provides two solutions: 1xEV-DO and 1xEV-DV. 1xEV-DO (1x evolution with data only) supports a theoretical peak data speed of up to 2.457 Mbps. and 1xEV-DV (1x evolution with data and voice) offers peak rate up to 3.072 Mbps.

- **TD-SCDMA**. Time Division Synchronous CDMA (TD-SCDMA) is the combination of the advanced TDMA/TDD (Time Division Duplex) system with an adaptive CDMA component for synchronous operations. TD-SCDMA was incorporated by the 3GPP (3rd Generation Partnership Project) [91] as part of the UMTS Release. Siemens mobile has developed this solution, which will be deployed in China. TDMA uses a 5 msec frame for repetitive transmissions and the frame is subdivided into 7 time slots. TDD principles permit traffic to be uplink and downlink using the same frame and different timeslots. For asymmetric services used with Internet

access, more time slots are used for the downlink than the uplink. For symmetric services used during phone calls and teleconferencing service, the same amount of data is transmitted in both direction. The data rates may range from 1.2 Kbps to 2 Mbps depending on the type of applications.

2.3.5 4G/B3G networks

Systems beyond third generation (B3G or 4G) [92, 93, 71, 94, 95] may be characterized by a horizontal communication model, in which several access technologies such as cellular, coreless, WLAN type system, systems for short range connectivity and wired systems will be combined into a common IP-based and common platform to complement each other. The technological trends beyond 3G may be both the seamless roaming and handoff between several air interfaces and the continuous development of the existing air interfaces. There are several technical issues, such as medium access, handoff, location coordination, multicasting, and QoS support, as well as economical aspects (pricing and billing), which must be researched and resolved. While the services provided in the 3G will probably be available and improved in B3G, the services that are available in wired systems may become available in future systems.

2.4 Bluetooth

Bluetooth (IEEE 802.15) [96, 97, 3, 7, 98] is a personal area wireless networking standard for low-cost, short range radio links between mobile PCs, mobile phones and other portable devices. Bluetooth radio technology provides a universal bridge to existing data networks, a peripheral interface, and a mechanism to form small private *ad hoc* groupings of connected devices away from fixed network infrastructures. The system operates with spectrum spreading accomplished by Frequency Hopping in ISM band starting at 2.402 GHz and stopping at 2.480 GHz. The data rate is limited to about 1 Mbps. The maximum userrate that can be obtained over the asynchronous link is 723.2 Kbps. The synchronous links link supports a full-duplex link with a user rate of 64 Kbps in both directions.

Bluetooth technology provides the multiple *ad hoc* connections sharing the same medium at the same location. Many independent networks overlap in the same area indicated as a scatter *ad hoc* environment. Each *ad hoc* network is called a *piconet* and multiple piconets in the same area are refered to as a *scatternet*. A piconet contains one master and several slaves. The number of units that can participate on a common channel is deliberately limited to eight (one master and seven slaves) in order to keep a high-capacity link between all the units.

Bluetooth adopts FH-CDMA. In the 2.45GHz ISM band, a set of 79 hop carriers have been defined at a 1MHz spacing. The channel is a hopping channel with a hop dwell time of $625\mu\text{s}$ which corresponds to a single slot. A large number of pseudo-random hopping sequences have been defined. The particular sequence is

determined by the unit that controls the FH channel, which is called the *master*. All other participants on the hopping channel are *slaves*, which use the master identity to select the same hopping sequence and add time offsets to their respective native clocks to synchronize to the frequency hopping. The master unit selects the hopping sequence whose cycle covers 23 hours, and 32 consecutive hops span about 64MHz spectrum.

An FH Bluetooth channel is associated with a piconet. Each Bluetooth radio unit has a free-running system or native clock. When a piconet is established, the slaves add offsets to their native clocks to synchronize to the master. Each unit can become the master that controls the traffic on the piconet and takes care of access control to avoid collisions. The master/slave role is only attributed to a unit for the duration of the piconet. In the master transmission, it includes a slave address of the unit for which the information is intended. For each slave-to-master slot, the master applies a polling technique deciding which slave is allowed to transmit. Only the slave addressed in the master-to-slave slot directly preceding the slave-to-master slot is allowed to transmit in this slave-to-master slot.

The Bluetooth system chooses packet-based transmission. The information stream is fragmented into packets. In each slot, only a single packet can be sent. On the slotted channel, synchronous and asynchronous links have been defined. The SCO (Synchronous Connection-Oriented) link is a point-to-point link between the master and a single slave. On this link, single-slot, three-slot, and five-slot packets are available. The ACL (Asynchronous ConnectionLess) link is a point-to-multipoint

link between the master and all the slaves in the piconet. Only a single-slot packet has been defined. This link supports a full-duplex link.

Because Bluetooth uses packet-based communication over slotted links, it is able for a communication unit to interconnect different piconets and to participate in different piconets in a scatternet. However, since the radio can tune to a single hop carrier only at any instant in time, a unit can communicate in one piconet only. However, the unit can jump from one piconet to another by adjusting the piconet channel parameters (i.e., the master identity and clock). A unit can also change roles when jumping from one piconet to another, but a unit cannot be the master in different piconets. The key is the hop selection mechanism designed to allow for interpiconet communications by changing the identity and clock to the selection mechanism, instantaneously a new hop for the new piconet is selected. In Bluetooth, a HOLD mode is used to allow a unit to temporarily leave one piconet and visit another.

2.5 CHUM related work

The authors in [99] propose an approach to share WWAN connection in an *ad hoc* network with participating hosts. They assume that all hosts have a WLAN interface and some hosts have a WWAN connection. The proposed mechanism discusses how a host that has no WWAN connection may share the idle bandwidth of a WWAN link of one of their cooperating hosts using the shared WLAN connection. The host without a WWAN connection transmits a message on a common WLAN multicast

address requesting the current traffic status of hosts having WWAN link. Any host that has WWAN connection measures its current WWAN traffic load and replies the measurement. The requesting host receives the measured traffic information from several hosts with the WWAN link and selects the least loaded gateway host. The gateway host performs an admission control of the connection request depending on the current bandwidth utilization of WWAN. Their research has different assumptions from the CHUM network. In their study, all hosts are not equipped with a WWAN connection. Their mechanism did not consider security issues, freeriding, failure recovery, and telecommunication costs. In addition, the host with the WWAN is treated as a gateway to outside networks, but the data downloaded is not shared by other hosts.

The CHUM concept has been previously designed for instant messaging. Zhu and Mutka [100] proposed a credit-based fair sharing of the proxy role in a CHUM network. The credit is evaluated from a peer's own experience as well as from other peer's recommendation. As a peer is served by a proxy, the proxy decreases the credit of the peer. The proxy stops serving a peer if it has no credits remaining. In order to earn credits, a peer must serve as a proxy. In [4] they use a proxy to indicate message presence information. All participating peers share a single message notification channel of the proxy. While peers turn off their WWAN connection for most of the time in order to save battery power, the proxy keeps the message arrival events and notifies the presence of the messages waiting. Any peer receiving the

notification contacts its associated instant messaging server and retrieves the message using its own 3G connection.

The CHUM explores ideas related to peer-to-peer computing. Some popular peer-to-peer applications have come from the file sharing programs such as Napster, Gnutella [101], and Jungle Monkey [102]. PAST [103] and Chord [104] provides anonymous peer-to-peer file sharing. Freenet [105] and Publius [106] allows peer-to-peer file systems that enable anonymous storage service. These systems use encryption, probabilistic routing or secret sharing schemes to guarantee clients and publishers' anonymity. Mojo Nation [107] supports file sharing mechanism with accounting facilities to identify which peers have contributed to the file sharing facilities, from which micro-payment schemes might be developed. The Free Haven Project [108] has anonymous sharing and an accountability scheme that has the goal of thwarting denial of service attacks, in which those who contributed positively to the system will have increased reputation.

Peer-to-peer media streaming is one of the interesting topics in peer-to-peer computing on the Internet. End System Multicast (ESM) [109] provides a multicast service to member hosts on the Internet without support of the IP level assistance in the routers. All functionality including group management and packet replication is pushed to the end hosts actually participating in the multicast group. ESM is suitable for small sized group of hosts exchanging multimedia contents such as video conferencing.

The authors in [110] investigated the performance improvements of the multiple description (MD) Content Delivery Network (CDN) over the single description (SD) CDN. MD coding is a compression scheme where a signal is coded into independent bit streams, referred to as descriptions. When any one description is received, a receiver is able to decode baseline quality content, and the additional description improves the quality of the content. The path diversity of multiple sources in MD-CDN has an advantage of reducing the probability of simultaneous loss in the paths. The paper concludes that the multiple description shows better performance of packet delivery in existing infrastructures.

PRISM [111] uses IP as the fundamental infrastructure to distribute streaming media using multicast over the IP network. This architecture allows accessing both the contents on the scheduled airing time and the contents stored with the network which allows on-demand access. Users retrieve content based on both the name of the content and the time it was aired.

Overcast [112] is an application-level multicasting system that allows data to be sent once to many destinations. Overcast provides data delivery for both live data and stored on-demand data. Clients are able to specify a starting point of an archived data such as the beginning or five minutes ago in order to catch up online content.

Tran, et al [113] focused on reducing the end-to-end delay of media streaming from a single source to multiple clients on the Internet. The authors propose a scheme that builds a client multicast tree by restricting the height and the node degree of the tree, which reduces the number of processing hops along the path. The failure

recovery process is performed regionally in order to minimize the number of the affected clients and to avoid the burden to the source. The authors in [114] study the assignment of multiple streaming sources with different outbound bandwidth offers to one requesting peer. The paper assumes that the media data may be partitioned into small sequential segments. They suggest an algorithm to assign a subset of the media data to each supplying peer, expecting minimum buffering delay in the streaming session.

SplitStream [115] allows a source peer to split a file into k stripes and to multicast each stripe using k multicast trees. Each member peer becomes both a non-leaf node in some of the multicast trees so that it forwards the part of the content, and a leaf node in some trees only to receive the striped content. SplitStream builds k multicast forests in order to spread the forwarding load across all participating peers.

In BitTorrent [116], when many downloaders try to download a file(s) from a URL-specified location, the downloaders upload to each other concurrently to help reducing the load of the source. BitTorrent partitions files into pieces of fixed size, typically 256 KBytes, and keeps track of which downloaders have what pieces.

Chapter 3

One-Tier CHUM Network

3.1 Introduction

A CHUM network is an *ad hoc* network in which mobile devices cooperate to reduce connection costs when accessing wireless global networks. When nearby mobile device users have been downloading the same multimedia streaming data, they may construct a CHUM network to share the WWAN bandwidth with other participating users, which is illustrated in both Fig 1.2 and Fig 1.3 of chapter 1. This chapter explains an operation of a one-tier CHUM network, labeled $CHUM_1$, and how members of a $CHUM_1$ network cooperate in order to download multimedia data, such as on-air TV programs. Each peer in a $CHUM_1$ network takes turns serving as a proxy, which makes an Internet connection and downloads multimedia data for other members. Then, the proxy distributes the data to other peers using its *ad hoc* connection, which is called *chumcast*. The proxy maintains and updates peer membership

information and determines the next $CHUM_1$ proxy. There are several challenges to develop multimedia chumcasting. First, $CHUM_1$ networks must be formed and effective delivery mechanisms must be used to distribute data to peers within $CHUM_1$. Since members are required to take turns serving as a proxy in order to share the WWAN costs, *freeriding* must be discouraged. However, since peers may join and leave $CHUM_1$ at any time, membership maintenance and failure recovery must be handled. Furthermore, if the proxy fails or leaves the $CHUM_1$ group without hand-off to another peer, recovery procedures must be established. This chapter addresses these challenges.

3.2 Network formation

The type of $CHUM_1$ network presented in this chapter is a special kind of an *ad hoc* network. All participating peers in the network agree to share the same multimedia content and take turns to become a proxy in order to download the agreed content and to broadcast packets to its neighbor peers. The remaining peers may decide to rebroadcast packets if they have neighbors that do not receive the packets from the proxy nor from the previous broadcasting peer. A $CHUM_1$ network does not maintain route information to reach any member peer, but rather all peers have the same group ID (GID) that indicates the membership of a specific $CHUM_1$ network. The GID assures that properly participating peers may have access to the shared multimedia data. Each member of the $CHUM_1$ network cooperates to deliver packets from the proxy to all other peers that have the same GID. Only the peers with the same GID

may forward and exchange packets with each other. In addition, each peer keeps two-hop neighbor information, as is done in other broadcasting schemes [65, 67, 66]. The two-hop information may be achieved by periodic exchange of HELLO packets in which a list of discovered neighbors of the sender is included. Together with the GID and two-hop neighbor information, a $CHUM_1$ network may form a mesh of peers, and data packets flow from a proxy to all its member peers.

Before forming or joining a $CHUM_1$ network, any mobile device first initiates its 3G connection and makes it ready for access to the Internet. The mobile device may not have a unique IP, because its ISP may provide a network address translation (NAT) [117] service to access the Internet. The proxy will assign a peer ID (PID) for each joining peer that is a unique network address within a $CHUM_1$ network. Address allocation for the *ad hoc* network is outside the scope of this chapter, but is discussed in [118]. When a mobile device is ready to connect to an outside network via its ISP, it looks for the service-provision beacon from a proxy to determine whether there exists an appropriate nearby $CHUM_1$ network. The beacon includes the description of current multimedia content such as the URL and/or a brief summary, as well as the proxy's PID. If the device does not find a proxy after some number of trials, it uses its own 3G connection to form a single-member $CHUM_1$ network, and becomes a proxy that will periodically broadcast a service-provision beacon. Since the proxy has no neighbors at this moment, data packets will not be broadcasted.

Suppose a mobile device finds a wanted proxy. It performs the join procedure and becomes a peer to the proxy. The first step is to send a "JOIN" packet to the

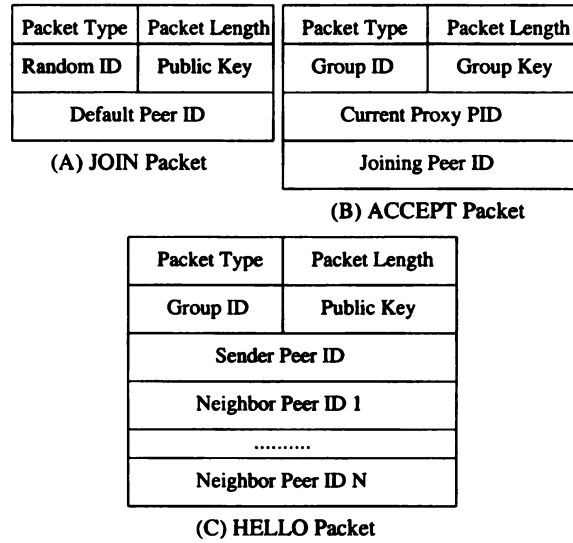


Figure 3.1: JOIN, ACCEPT, and HELLO Packet Format

proxy. The contents of the packet is depicted in Figure 3.1(A). Since the device does not have a PID of a $CHUM_1$ network, it uses a default PID. In addition, the device generates a random number that distinguishes itself from other joining devices. For security reasons, the JOIN packet includes the sending device's public key. The proxy receiving the packet responds with an "ACCEPT" packet in which the group information is encrypted with the public key of the joining peer. Figure 3.1(B) shows the format of the packet. In case that the joining peer receives the ACCEPT packet, it configures itself with the PID in the packet for its network address in the $CHUM_1$ network and stores the current GID, group key and PID. All peers in a $CHUM_1$ network share the same GID. Only the peers with the same GID exchange data and control packets, and discard them if the GID in the packet is different from their own. To prevent freeriders who receive services but do not participate, the proxy encrypts

all $CHUM_1$ -wide control and data packets. All peers require a symmetric group key in order to decrypt packets sent by a proxy.

When a proxy serves a $CHUM_1$ network for a fixed amount of time and there is at least one peer in the network, the proxy role will be moved to another peer. In one-tier $CHUM_1$ networks, the proxy maintains a circular scheduling list of peer members. Other methods for fair sharing of the proxy role are possible, but no further study is developed here. When a peer becomes a proxy, the next peer in the scheduling list becomes a vice proxy, which will be the next proxy. The current proxy receives scheduling information from the previous proxy. If a change in membership occurs, the current proxy notifies the change to the vice proxy. That is, the proxy and the vice proxy share the same membership information. If a new peer is introduced, the peer is placed after the vice proxy in the scheduling list. The reason is that new members should serve soon as a proxy to ensure that they appropriately participate before benefiting from the service of other peers.

3.2.1 Neighborhood maintenance and data chumcasting

Every peer maintains its two-hop neighborhood information, which is similarly used by other broadcasting algorithms [65, 67, 66]. The two-hop status may be achieved by exchanging “HELLO” packets. Figure 3.1(C) displays the items in the packet. When a peer receives this packet, it examines the GID in the packet. Only the peers with the same GID will process and exchange HELLO packets as well as keep the neighbor information. The peer includes its PID in the packet that is unique within

the $CHUM_1$ network. The PID is required when a peer needs to do one-hop unicast to its neighbors individually. Neighbor PIDs in the packet provide knowledge of the two-hop neighbors. One-hop neighbors are the peers that send HELLO packets and two-hop neighbors are the peers whose PIDs are in the packet excluding itself.

A $CHUM_1$ network utilizes a broadcast mechanism to deliver packets to all peer members using two-hop information. Many broadcasting algorithms in *ad hoc* networks try to improve their performance by dealing with two inherent problems: packet duplication and collision. Collision in broadcasting is very critical when a sender broadcasts a packet and all remaining members rebroadcast it to their neighbors at the same time. Many collisions are expected to occur in wireless *ad hoc* network, which is referred to the *broadcast storm problem* [61] in flooding. $CHUM_1$ networks contain mechanisms to manage these problems. When a proxy sends a data packet, it attaches a serial packet number in order to distinguish the duplication of packets. Many broadcasting algorithms use a random time interval, called Random Assessment Delay (RAD), which is selected from a uniform distribution between $(0, max)$, where *max* is the largest delay interval. Any peer that needs to rebroadcast must wait until the random amount of time. The duplicated packets received during the waiting time will be dropped. RAD decreases collision of packets as well as reduces duplication.

One important broadcasting algorithm issue is to minimize the probability of collision of packets by properly selecting the rebroadcasting nodes in the network. Williams [59] categorizes broadcast protocols into several domains such as simple

flooding, probability-based methods, area-based methods, and neighbor-knowledge methods. The authors conclude that neighbor-knowledge methods are preferred over other types of broadcast protocols based on their simulation results as well as the power ramification. The Scalable Broadcast Algorithm (SBA) [65] is a broadcasting protocol that uses 2-hop knowledge. The main idea of the algorithm is that a node will rebroadcast a packet if all of its neighbors have not received the packet by the previous transmission. Figure 1.3 shows one example of this broadcast protocol. Suppose proxy B broadcasts a packet to its neighbors A, C, and D. Peer A knows that B and C are its neighbors and B's transmission covers both peers, and concludes that no retransmission is required. In contrast, peer D has four neighbors, B, C, E, and F. B's transmission reaches B's neighbors, A and C. It implies that peer D needs to retransmit the received packet for E and F. Peer F executes the same steps and rebroadcast to its neighbor G.

3.3 Membership management

When a peer joins a $CHUM_1$ network, the proxy inserts peer information into its scheduling list. As long as the information of the peer is in the list, the proxy assumes that the peer is still in the network. Each item in the list contains a timer associated with a peer. When the timer expires, the proxy recognizes that the associated peer leaves the network without an explicit notice. Any peer that wants to stay to receive continuous service must send a periodic REFRESH packet to its proxy. The peers in between the proxy and the refreshing peer should help forward the REFRESH packet

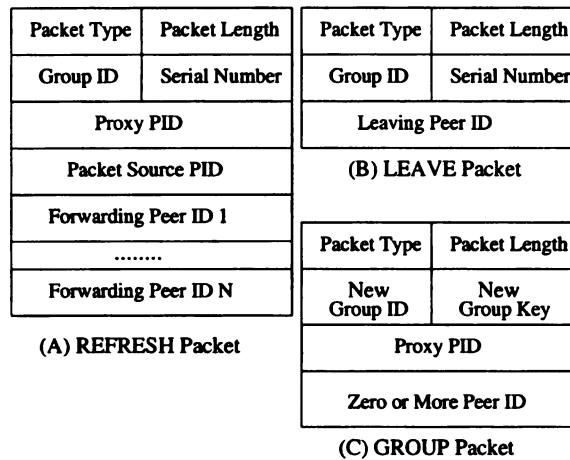


Figure 3.2: REFRESH, LEAVE, GROUP packet format

to the proxy. The forwarding peers append their PID at the end of the packet, which may help detect a duplicate as well as provide path information that may be used by the proxy. If the proxy receives a REFRESH packet, the timer of the associated peer will be reset. In addition, the proxy may collect route information to any peer, especially the vice proxy, by receiving a REFRESH packet from the peer.

Figure 3.2(A) displays the format of a “REFRESH” packet. The proxy PID is the PID of the current proxy acquired from the ACCEPT packet. The packet source PID is the PID of the packet initiator that wants to refresh its membership. The combination of the packet source PID and the serial number uniquely identifies a packet. When a peer receives a REFRESH packet from its neighbor, it examines the packet source PID and the serial number to determine whether the same packet was seen before. If it is a duplicate, the peer drops the packet. If not, the peer stores the PID and the number. In addition, it further examines whether there is any neighbor

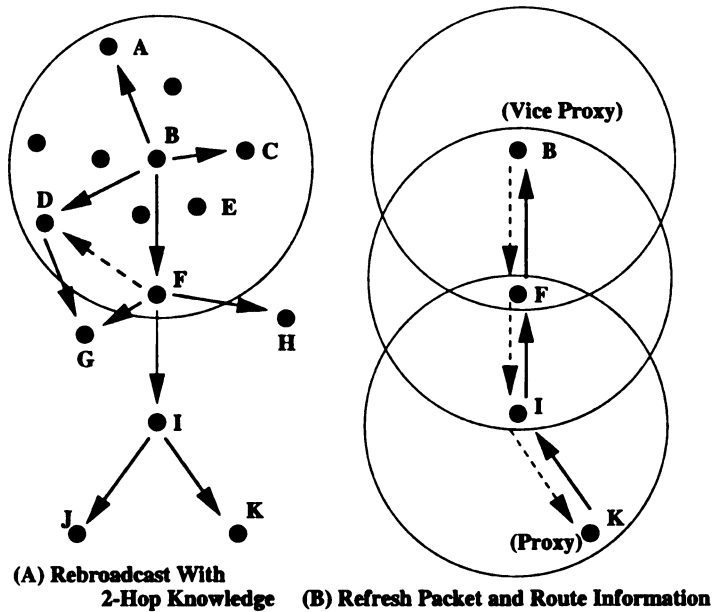


Figure 3.3: Packet Delivery and Route Discovery

uncovered by the previous transmission using two-hop knowledge. In Figure 3.3(A), several peers including A, C, G, H, and J are unnecessary to rebroadcast and drop the packets because they have no such neighbors. If a peer has these neighbors, it appends its PID at the end of the REFRESH packet and updates the packet length properly. Then, it rebroadcasts the modified packet to its neighbors. If the proxy is one of its neighbors and it is covered by the previous transmission, no retransmission is expected. When the packet reaches the current proxy, the associated timer is reset. Suppose a peer B wants to send a REFRESH packet to a proxy K in Figure 3.3. Then the PID of F and I will be appended at the end of the packet. While the proxy K receives the packet, it now knows that the backward route in the packet is the route to reach the peer, displayed in Figure 3.3(B). Suppose the REFRESH packet sender

is a vice proxy, then the proxy is able to send membership information to the vice proxy using this route information. The mechanism to find a route works similar to that of DSR [28] in which a source node sends a route request packet RREQ and the route information comes back with a route reply packet RREP.

Any peer may leave a $CHUM_1$ network with or without notice to a proxy. Leaving the network without notice may trigger a timeout by the proxy. In contrast, a peer may send an explicit notice by transmitting a “LEAVE” packet in Figure 3.2(B). The transmitting and rebroadcasting mechanisms are the same as that of the RE-FRESH packet. When a proxy receives the packet, it removes the associated peer information from its scheduling list. The vice proxy must be notified of this event with the known unicast route.

When a proxy detects a peer leaving the $CHUM_1$ network due to its timer or by receiving a LEAVE packet, the proxy creates a “GROUP” packet and delivers it to all its member peers. The content of the packet is shown in Figure 3.2(C). The purpose of this packet is to exclude the peer that left the network by replacing the GID and the group key. When both GID and key are determined, the proxy packs them in the packet with its PID and the PID of the leaving peer. The packet will be delivered in an one-hop unicast fashion encrypted with each neighbor’s public key. The two-hop knowledge helps a peer determine which peer should not receive the GROUP packet. If a forwarding peer finds that the peer that left is its one-hop neighbor, the peer will not send the packet to the neighbor.

3.4 Recovering from failures

A $CHUM_1$ network operates in a highly mobile environment, which implies that any peer may leave the network without notice to its proxy, and even a proxy itself may fail or disappear at any time. Moreover, some rebroadcasting peer(s) may leave, resulting in partitioning one $CHUM_1$ network into two or more isolated subnetworks. The subnetwork(s) without a proxy may not receive data. The $CHUM_1$ network should deal with these problems properly and efficiently.

3.4.1 Recovery of a missing peer

Any peer may leave the network at any time and the absence may be detected by a timeout mechanism. The information of the peer that left will be deleted from the scheduling list maintained by a proxy. The update event will also be informed to the vice proxy so that the proxy and the vice proxy maintain consistency of the scheduling list. Additionally, the proxy picks a new GID and a group key, and distributes them with a GROUP packet. The packet implies that the membership changes are due to an absent peer, and the remaining peers will use a new GID and key for upcoming packet exchanges.

A vice proxy may leave its network without sending its periodic REFRESH packet. When the proxy recognizes that its vice proxy is missing, it sends a new GROUP packet to all peers, and appoints a new vice proxy that is located next to the missing vice proxy in the scheduling list. The proxy forms a "SCHEDULE" packet

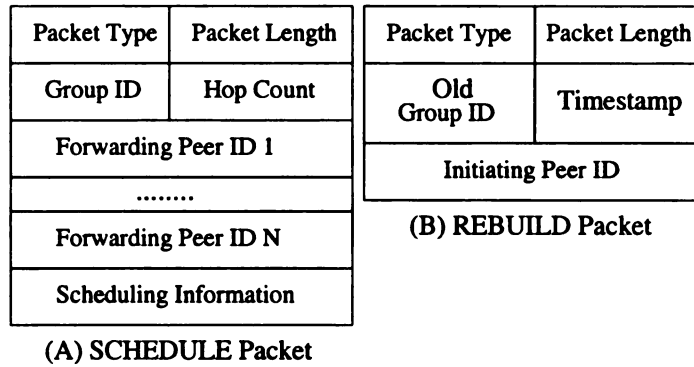


Figure 3.4: SCHEDULE and REBUILD Packet Format

shown in Figure 3.4(A). The packet is constructed when a REFRESH packet arrives from the newly selected vice proxy. The route and the number of hops from the proxy to the vice proxy is inserted in the packet and the scheduling list is appended at the end of the packet. This packet is not broadcasted throughout the network, but only the peers whose PIDs are listed in the packet will forward it.

3.4.2 Proxy failure recovery

The most disastrous failure in a $CHUM_1$ network may be the failure of the proxy, because the proxy is the only source delivering data packets from the outside networks. The vice proxy, which is appointed as soon as the proxy is determined, will be the most prominent candidate to be the next proxy when the current proxy fails or disappears. The vigilant vice proxy keeps a proxy timer that is reset when a data packet arrives from the current proxy. If the proxy timer expires, the vice proxy assumes the proxy has failed or left the network, and prepares a GROUP packet.

When the vice proxy generates a new GID and key, it adds them into the packet and the prior proxy's PID is specified as the peer that left. This procedure is similar to the proxy handover from a proxy to a vice proxy except that the leaving peer's PID will be empty. When a peer receives a GROUP packet with no leaving peer's PID, it understands that proxy handover has occurred. If the PID part is not empty, the peer knows that membership change has occurred. Any peer updating the new information may refresh its membership to the new proxy. When a REFRESH packet arrives from the newly assigned vice proxy, the proxy shares its membership information with the new vice proxy using a SCHEDULE packet. The proxy will connect to the same content provider and resumes the chumcast in order to serve the network.

3.4.3 Network partitioning recovery

It may happen that both the proxy and the vice proxy do not operate. One possible case is that both leave or fail at the same time. Another case may be that a $CHUM_1$ network is partitioned into two or more subnetworks because some forwarding peers between the proxy and other peers moved and data packets from the proxy cannot be delivered to peers in the partitioned subnetwork(s). Each peer maintains a data timer that expires when no data packets arrive. The duration of the timer should be longer than that of a proxy timer maintained by a vice proxy. A peer whose data timer expires initiates $CHUM_1$ network recovery. The initiating peer generates and broadcasts a "REBUILD" packet that includes the group key that is used by

the former proxy. In addition, the packet contains a timestamp when this packet was created and the peer's PID, which is still unique within a partitioned $CHUM_1$ network. Figure 3.4(B) shows the format of the REBUILD packet. The packet is encrypted using the old group key.

When a peer receives a REBUILD packet, it looks at the GID in the packet and compares the GID with its stored value. If the two GIDs are identical, it stores the timestamp and the initiating peer ID, then the packet is rebroadcasted to its neighbors using two-hop knowledge. Suppose the peer receives another REBUILD packet from another initiator. It compares the GID and the timestamp in the packet with its stored values. If the GIDs are the same and the recently received packet has an older timestamp than that of the stored information, the peer updates the stored information with the older timestamp and its initiating peer ID. The clocks of all peers do not need to be synchronized and the packet with the oldest timestamp wins throughout the isolated network.

After a random time from the last update of the REBUILD packet, a peer constructs a REFRESH packet with the initiating peer's PID as a proxy PID and the old GID. If the serial number is set to all 1's, this implies that the packet is not for refreshment of a membership but for rebuilding a $CHUM_1$ network. All peers usually process packets with the same GID except the REFRESH packet with all 1's in the serial number. In case that this REFRESH packet is received, a peer compares the PID of its proxy rather than the GID. If the two PIDs are the same, the peer appends its PID at the end of the packet. Then, it modifies the packet length and

rebroadcasts the packet to its neighbors. Finally, when the initiating peer receives the REFRESH packet, it registers the sender of the packet as well as its rebroadcasting peers, and constructs a GROUP packet.

The initiating peer becomes a proxy and generates a new GID and a group key. At this time, the peer ID in the GROUP packet contains at least one PID: REFRESH packet sender, and zero or more refresh-packet-relaying peer(s). In order to prevent disclosing the GID and key to other peers that are not yet registered, the proxy encrypts the group information with the public key of the first peer listed in the packet. The public key of its neighbor is known when HELLO packets are exchanged. The proxy, then, transmits the GROUP packet and the first peer in the list may recognize the packet. After storing the group information, the peer encrypts with the public key of the second peer in the list, if exists, and retransmits it. Finally, the packet delivery process finishes when the refresh-initiating peer receives the new group information. The refresh process should be completed within the period of the initiating proxy because the REFRESH packets with all 1's in the serial number will compare the current proxy's PID. The current proxy should appoint a new vice proxy and share the membership information. The proxy will resume data service after delivering the first GROUP packet.

3.5 State transitions

The previous descriptions explain how one mobile device connects to the Internet using its 3G connection and shares it with other peers by cooperating with each

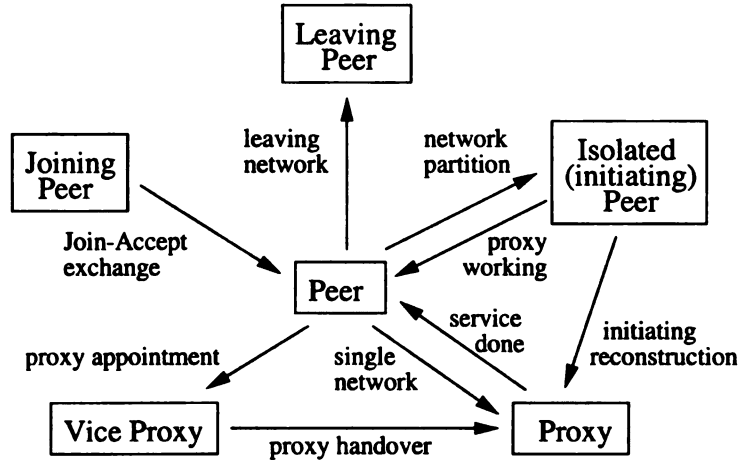


Figure 3.5: State Transition of a $CHUM_1$ network

other. Figure 3.5 summarizes the state transition of a mobile device in a $CHUM_1$ network. The *joining peer* is a mobile device that wants to participate in a $CHUM_1$ network and the *leaving peer* is a peer that is about to leave from a $CHUM_1$ network. When a mobile device becomes a peer member, it may play a role of a vice proxy and then a proxy in order to serve a $CHUM_1$ network. In some cases, a peer becomes an isolated peer, which has no proxy to serve. One of the initiating peers in a partitioned network takes a proxy position and serves other peer members in its rebuilt network.

All packets exchanged in a $CHUM_1$ network are displayed in Figure 3.6. In this figure, a proxy entity is placed at the center rather than a peer entity in the previous figure. All packets are exchanged between two different entities except the HELLO packet, which is swapped between neighbor peers. When an isolated peer receives a REFRESH packet, it works as a proxy. The transition to a proxy entity is marked as a dotted line.

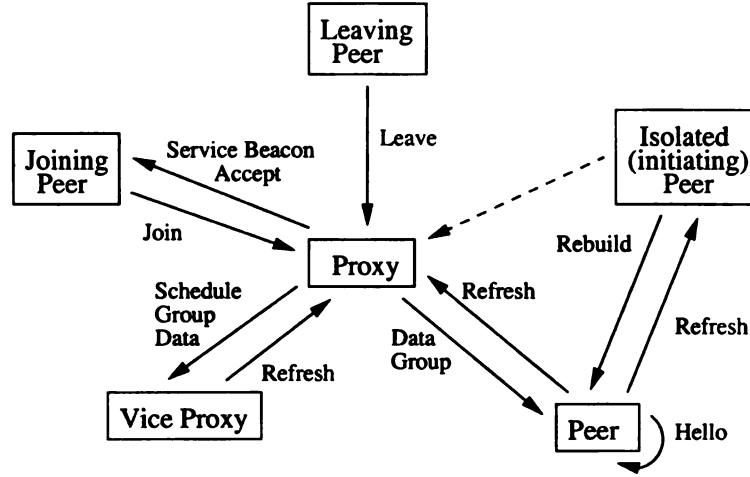


Figure 3.6: Packets Exchanged in a $CHUM_1$ Network

3.6 Summary

This chapter proposes an approach for mobile devices, called peers, to reduce the telecommunication cost by sharing the connection from one of the peers, called the proxy. All peers take turns being a proxy. Chumcast delivers packets to all connected members in a $CHUM_1$ network with a wireless broadcast mechanism. The peers in a $CHUM_1$ network consent to share the same multimedia data from a content provider (CP), and the proxy contacts the CP to download the data and chumcasts the data to its members. Each peer maintains two-hop neighbor knowledge in order to decide whether to rebroadcast packets. The proxy updates peer membership and session information periodically. The vice proxy helps recovery from the proxy failure. A partition of a $CHUM_1$ network may be detected and the peers in the isolated network(s) cooperate to recover.

Chapter 4

Two-Tier CHUM Network

4.1 Introduction

The CHUM network provides a content sharing scheme in which mobile peers can share their content with other participating peers in order to reduce telecommunication costs. In the one-tier CHUM network, labeled $CHUM_1$, the proxy maintains peer membership information and determines the next proxy, as well as downloads multimedia content from the Internet. The $CHUM_1$ network can be extended to cooperate with wired networks and wireless networks. The cooperative *ad hoc* multimedia CHUM network is labeled $CHUM_2$ network. The cooperation gives benefits to all participating peers such as the reducing workload of peers, the savings of wireless bandwidth, and the reduction of battery power. The benefits come from distributing some of peers' computation and communication workload to the wired networks.

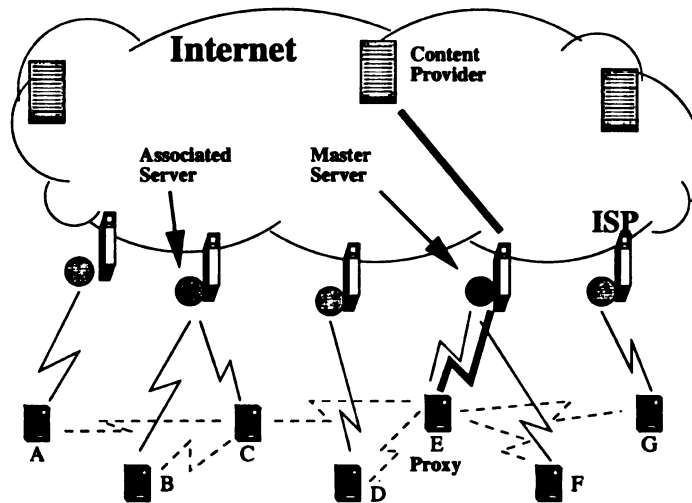


Figure 4.1: A single peer incurs telecommunication charges. The connection is shared by all peers.

Both Fig. 4.1 and Fig. 4.2 illustrate the cooperation between the mobile peers and the associated servers in the Internet. This figure includes additional components compared with Fig 1.2 or Fig 1.3, such as the associated servers, the master server, and the communication between the servers and the mobile peers. Fig. 4.1 shows that the devices downloading the same content construct a $CHUM_2$ network and one of them, say E, becomes a *proxy*. As the proxy E receives data from its WWAN connection, it broadcasts the data to its neighbor peers with its WLAN. The master server selects some of the participating peers, for example C, to rebroadcast the packets for other peers that are out of transmission range of the proxy, which was decided by each peer in one-tier networks using two-hop neighborhood information. Each peer has its associated server that may be located somewhere on the Internet such as at an Internet Service Provider (ISP), Content Provider (CP), or somewhere else. A server

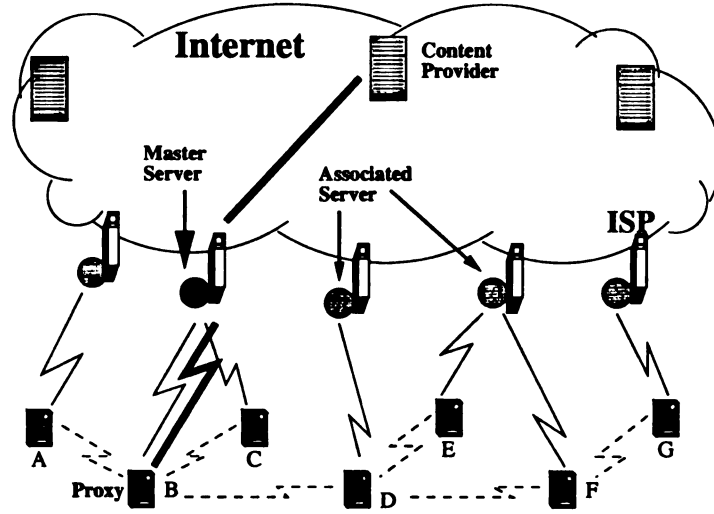


Figure 4.2: The proxy migrates to another peer

may be associated with several peers. For some cases, all peers may connect to a single server that controls the operation of the $CHUM_2$ network and deals with any license issues and fees with CPs. Each associated server collects neighbor information from its associated peer and reports it to the master server, which is the associated server of the proxy. The master server maintains membership information of a $CHUM_2$ network and the global topology of the logical $CHUM_2$ network, which allows the master server to schedule the next proxy and to compute the minimum rebroadcasting set of peers.

After a proxy serves for a given amount of time, the master server schedules another peer to serve as a proxy, which was performed by the current proxy in an one-tier network. The master server delivers all membership and topology information to the associated server whose associated peer is appointed as the next proxy. The associated server becomes the new master server and manages the $CHUM_2$ network.

Fig. 4.2 shows that peer B becomes the next proxy. The master server finds peer D and F as the minimum rebroadcasting set, and reports the result to the associated servers of peer D and F. The associated servers direct their peers to rebroadcast when data packets are received from the proxy or from one of its neighbor peers. In Fig. 4.1 and Fig 4.2, the thick connection line indicates that the multimedia data packets are downloaded from a CP. The thin line of the connection implies intermittent control packet exchanges between associated servers and their associated peers. The dashed line shows the packet exchanges over the *ad hoc* network. The proxy role will be transferred periodically to another peer that starts downloading from the same CP and broadcasts the data to all member peers in order to provide continuous service seamlessly. By sharing the connection and rebroadcasting downloaded data to other peers, the $CHUM_2$ network reduces the connection costs for all participating peers.

There are several challenges to develop two-tier low-cost multimedia chumcasting. First, the membership information of the mobile devices should be maintained in the $CHUM_2$ network. Because all participating peers should serve as a proxy at a time and some of them should rebroadcast the received packets, the master server needs to keep the current membership information of the *ad hoc* network. Some mobile peers may join or leave at any time, and the master server should update the membership changes. Second, the $CHUM_2$ network should guarantee that only the participating peers will receive the benefits of sharing contents, and any non-member devices, called *freeriders* should be discouraged to access the contents. The peers in

the network needs to share the same group ID in order to interact with the valid member peers.

Third, the content distribution from the proxy should be limited to the member peers correctly and securely. It is possible for some freeriders to overhear the contents, or some peers may attack a $CHUM_2$ network. The $CHUM_2$ network should be free of eavesdropping and robust enough to protect against attacks from inside the network and from outside. Fourth, the $CHUM_2$ network should be resilient enough to recover from several failures such as peer failures and the *ad hoc* network partition. Because the $CHUM_2$ network operates in a highly dynamic environment, some peers may disappear or fail. This case triggers the change of the shared group ID because some of the missing peers may become freeriders. In addition, if some intermediate peers fail, the *ad hoc* network may be partitioned and no data will be transmitted between peers in two or more partitioned networks. The network partition should be detected and properly treated for the correct operation of the $CHUM_2$ network.

4.2 Network formation

The $CHUM_2$ network is based on the cooperation of a wired network and a wireless *ad hoc* network. One of the most distinct differences between a $CHUM_1$ network and a $CHUM_2$ network is the existence of the *associated server*. Each peer in the wireless network associates with a server in the wired network. The associated server interacts with its associated peer in order to collect neighborhood information of the peer. In addition, the server provides information to its peer such as group information of the

$CHUM_2$ network participated and the decisions of rebroadcasting packets or not. One of the peers, called the *proxy*, is responsible for downloading multimedia data and broadcasts them to its neighbor peers, which is similar to one of the proxy roles in a one-tier network. The server associated with the proxy is called the *master server* that performs many important tasks including peer membership management, the scheduling of the next proxy, the selection of rebroadcasting peers, and the detection of its wireless $CHUM_2$ network partition, which are performed by the proxy or the cooperation of peers in a $CHUM_1$ network. In addition, the master server generates a symmetric key with which the proxy encrypts all multimedia data packets, and all peers decrypt them with the key. The key is passed from the master server to all associated servers that deliver the key to their associated peers. This key distribution path makes the two-tier network more secure than that of the one-tier network. Each associated server maintains a public key of all other associated servers in a $CHUM_2$ network. All packets exchanged between associated servers are encrypted with the receiver's public key. The associated servers decrypt their received packets with their private keys.

The formation of a $CHUM_2$ network is different from that of an $CHUM_1$ network. The first step for a mobile device to participate in a $CHUM_2$ network is to search for an existing $CHUM_2$ network nearby. If the device fails to find a wanted service beacon from a $CHUM_2$ network, it connects to an associated server via its ISP and downloads the user's favorite contents from its content provider (CP). The associated server generates a random group ID (GID) for the $CHUM_2$ network, a

| | | | | | |
|----------------------|---------------|---------------------------|---------------|------------------|---------------|
| JOIN | Packet Length | ACCEPT | Packet Length | HELLO | Packet Length |
| PID of Beacon Sender | | Master Server IP and Port | | Group ID | |
| Default Peer ID | | NID | | Sender Peer ID | |
| Random Number | | Received Random Number | | | |
| (a) JOIN Packet | | (b) ACCEPT Packet | | (c) HELLO Packet | |

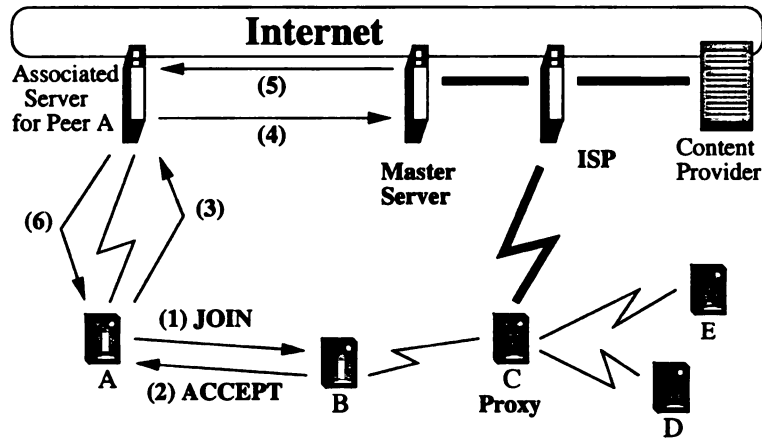
Figure 4.3: JOIN, ACCEPT, and HELLO Packet Format

network ID (NID) to distinguish between several $CHUM_2$ networks by the server, a unique peer ID (PID) for the device, and the symmetric group key for data packet encryption. The server delivers the group information, the NID, and the PID to its associated device. The device stores the information and periodically transmits a service beacon that contains a description of the $CHUM_2$ network including the URL of the CP, a summary of the downloading multimedia content, and the beacon sender's PID. In a single-member $CHUM_2$ network, there is no need for broadcasting the downloaded contents. If no peers join the $CHUM_2$ network until the termination of the CP, the network cost will not be reduced.

Suppose another mobile device looks for a service beacon and finds a $CHUM_2$ network that downloads interesting content from the Internet. The device constructs a JOIN packet and broadcasts it to nearby peers in the $CHUM_2$ network. The format of the packet is shown in Fig. 4.3 (a). Since the device does not have a unique PID in the $CHUM_2$ network, it uses a predefined default PID. In addition, the device generates a random number that distinguishes it from other joining devices. Each peer in a $CHUM_2$ network sends a service beacon periodically. Any peer receiving the JOIN packet compares the PID of the beacon sender in the packet with its own, and drops

it if unmatched. If both PIDs are identical, the peer replies with an ACCEPT packet displayed in Fig. 4.3 (b). The packet contains the transport information of the master server on the Internet as well as the received random number from the JOIN packet. When the device that transmitted the JOIN packet receives an ACCEPT packet with the same random number, the device becomes a peer of the $CHUM_2$ network. After examining the packet, the device contacts its ISP in order to communicate with an associated server on the Internet.

The device receiving the ACCEPT packet informs its associated server about the NID and the transport information of the master server. The associated server registers the presence of the new peer to the master server with the NID. As a response, the master server provides the group information of its $CHUM_2$ network to the associated server including the GID of the network, a shared symmetric group key for data packet decryption, and the PID of the joining peer, which is unique within the $CHUM_2$ network. The PID will be used as a network address of the newly joined peer in the *ad hoc* $CHUM_2$ network. The associated server delivers the $CHUM_2$ network information including the GID, the symmetric key, and the PID to its associated peer. The peer configures with the PID as a network address in the wireless $CHUM_2$ network, and stores the GID and the group key for data packet decryption. Device mobility does not produce any overhead during the CHUM network formation, because the initial proxy resumes broadcasting downloaded content after the active server collects neighborhood information. The peer movement during the formation does not matter, but it is important to collect the location of peers right



- (1) New device sends a JOIN packet to existing CHUM network
- (2) ACCEPT packet contains info about the master server
- (3) Peer delivers master server info to its associated chumcast server
- (4) Associated chumcast server registers its peer to master server
- (5) Master server passes GID and group key to associated server
- (6) GID and key passed to associated peer to join CHUM network

Figure 4.4: The sequence of introducing a mobile device to a $CHUM_2$ network

before the proxy resumes content transmission. The sequence of introducing a mobile device to a $CHUM_2$ network is displayed in Fig. 4.4. The associated server of peer A is located at the ISP, but peer C's server does not reside in the same place as its ISP. The servers for other peers such as B, D, and E, are not shown.

4.3 $CHUM_2$ network management

4.3.1 Membership management

All peers in a $CHUM_2$ network take turns to become a proxy in order to share the telecommunication costs among the participating peers. The proxy may be a place

to maintain all peer information and schedule the next proxy, which is true for a $CHUM_1$ network. However, it is more beneficial to keep the peer information in the master server for two reasons. First, it saves the wireless bandwidth because all membership information may be exchanged between associated servers in the wired network. Second, the proxy saves the battery power that might be consumed for the membership management and other required computation. In a $CHUM_2$ network, every peer gathers its one-hop, rather than two-hop, neighbor information that is delivered to its associated server periodically, and then to the master server. The master server draws a complete logical picture of the $CHUM_2$ network using the partial neighbor information from each associated server, and performs several crucial tasks for the $CHUM_2$ network.

When a peer joins a $CHUM_2$ network, the master server updates the membership information of all peers in the network. The peer information is used for scheduling the next proxy, selecting the minimum rebroadcasting set of peers, and detecting a network partition. The membership may change when a peer leaves the $CHUM_2$ network. The peer that leaves may explicitly inform its associated server. Nevertheless, some peer may disappear without any notice. The impolite disappearance may be detected by its associated server, which expects periodic neighbor information from its peer. When the associated server detects the missing periodic information, it promptly reports the fact to the master server so that it updates the membership information and recomputes the rebroadcasting set as well as any network partition. Moreover, the master server generates a new GID and group key, and

passes them to all associated servers, except the one whose associated peer has left the network. It is because the missing peer may become a freerider using the group key. The associated servers deliver the new group information to their peers. Upon receiving the information, each peer updates the GID and saves the group key. The HELLO packet that follows includes a new GID in order to exclude the left peer from the $CHUM_2$ network.

In order for the master server to perform its tasks, it requires neighbor information of peers in the $CHUM_2$ network. The neighbor information should be collected from each peer in the network. In order to discover the neighbors, each peer constructs a HELLO packet shown in Fig. 4.3 (c). The neighbors of a peer are the peers from which HELLO packets are received. Different from one-tier networks, HELLO packet does not include the neighbor peer IDs displayed in Fig. 3.1. Two-hop knowledge is required in order to determine whether to rebroadcast the received packets. In two-tier network, these decision making procedures are unnecessary. Only with one-hop information from each peer, the master server can simply build the global connectivity matrix of a $CHUM_2$ network and the minimum rebroadcasting set of peers with the global connectivity information.

When a peer receives a HELLO packet, it examines the GID in the packet. Only the peers with the same GID will process HELLO packets as well as keep the neighbor information. Fig. 4.5 presents how the neighbor information is collected and used in a $CHUM_2$ network. Periodically, each participating peer in a $CHUM_2$ network broadcasts a HELLO packet with its PID and GID. After a given time, a

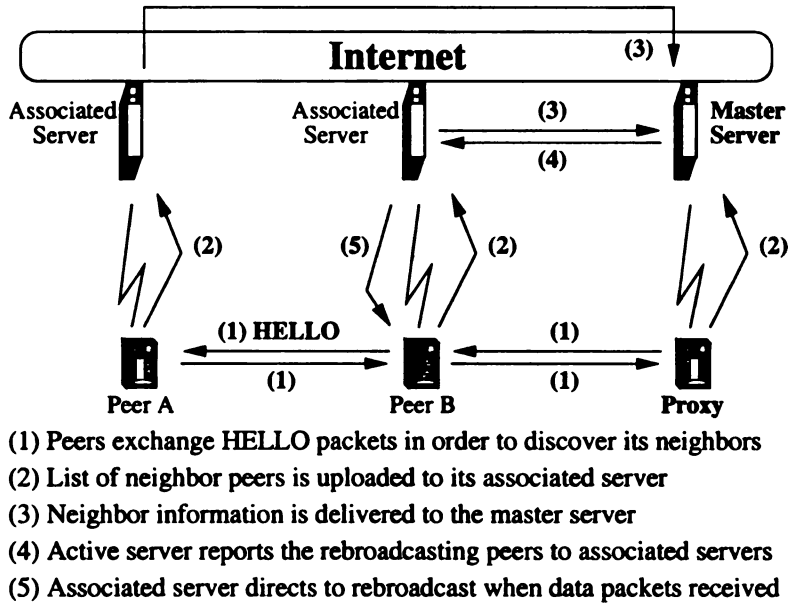


Figure 4.5: The sequence of processing neighbor information in a $CHUM_2$ network

peer is able to discover its neighbors and to generate a list of its neighbor peers that sent HELLO packets. The peer, then, uploads the list to its associated server, which delivers the list to the master server. The master server may create a global topology of its $CHUM_2$ network from synthesizing the partial neighbor information from each peer. Whenever the active server receives new neighborhood information, it updates the global topology of the network. The global topology is essential to compute the minimum rebroadcasting set of peers and the efficient partition detection of a $CHUM_2$ network. The master server maintains a boolean connectivity matrix in which an element of (i, j) is set to true if i^{th} peer is a neighbor of j^{th} peer.

4.3.2 Selection of rebroadcasting peers

The main job of the proxy is to contact its favorite content provider on the Internet, to download the multimedia content, and to broadcast the content to all peers in its $CHUM_2$ network. While the proxy downloads the content, it encrypts the content with the symmetric group key and broadcasts them to its neighbors. The peers that are far from the proxy may not receive the packet unless some peers in the *ad hoc* network rebroadcast them. Much research have proposed distributed algorithms for selecting the rebroadcasting neighbors while using local neighbor information [62, 66, 67, 119]. Such research focused on two-hop neighbor knowledge to minimize the number of rebroadcasting peers. Because wireless broadcasting may cause duplication of packets and the wireless medium contention, it is important to select carefully the set of minimum rebroadcasting peers in a $CHUM_2$ network.

One of the distinct advantages of the two-tier network over the one-tier network is the selection of the rebroadcasting peers. In an one-tier network, each peer maintains two-hop neighbor information and executes the decision algorithm whether to rebroadcast the received packet in a distributed manner when packets arrive. The peers in a two-tier network do not maintain any neighborhood information nor execute the decision algorithm. The minimum rebroadcasting set of peers is determined by the master server and the associated servers of the peers in the set are notified from the master server. Each notified associated server directs its associated peer to rebroadcast when the peer receives data packets from its neighbor. This mechanism allows for the peers to save the memory space and the computation power.

The problem of selecting the minimum number of rebroadcasting set is very similar to the MCDS (Minimum Connected Dominating Set) problem. The authors in [62] mentioned that the minimum rebroadcasting set problem is harder than the MCDS problem, which has been proved to be NP-complete. Several wireless broadcasting algorithms [62, 66, 67, 119] have been proposed for the approximation of MCDS to compute the backbone wireless peers or the rebroadcasting peers using local neighborhood information. Starting from the proxy, two-hop knowledge helps to decide the next rebroadcasting peers in a completely distributed manner.

Wireless *ad hoc* broadcasting algorithms seldom use the global topology of a network due to the lack of a centralized controlling peer and the scarcity of wireless bandwidth for exchanging neighbor information with all other hosts throughout the network. However, End System Multicast (ESM) [109], operating on a wired Internet, requires each host in ESM to maintain a global view of the network topology of the multicast members in order to find the minimum cost of total unicasts for a multicast packet. Because ESM is targeted for a small size, a few hundred hosts, and sparse groups such as video conferencing and virtual classroom, the bandwidth for exchanging neighbor information among the participating hosts will not be a problem.

The master server in a $CHUM_2$ network also utilizes the global topology of the $CHUM_2$ wireless network, and produces the minimum set of rebroadcasting peers. The best known algorithm for the MCDS problem is the Berman's algorithm [120]. The algorithm generates a set of connected nodes and then merges the set into one connected dominating set. The Berman's algorithm requires $O(D)$ steps and an ap-

proximation ratio of $\ln \Delta + 3$, where D is the diameter of the network and Δ is the maximum degree of the tree. ESM also has two steps to construct an overlay spanning tree for data delivery to all members. First, they construct a mesh of connected graph and then build the shortest path spanning tree of the mesh. In a *CHUM*₂ network, the partial neighbor information from each associated server provides a hint to generate a mesh of connected peers. The master server merges the connected peers into one connected dominating set of peers, which includes the proxy in the wireless *CHUM*₂ network. The master server notifies the associated server of the peers in the set, which is shown in step (4) of Fig. 4.5. Each responsible associated server directs its peer(s) to rebroadcast packets when it receives data packets from its neighbor.

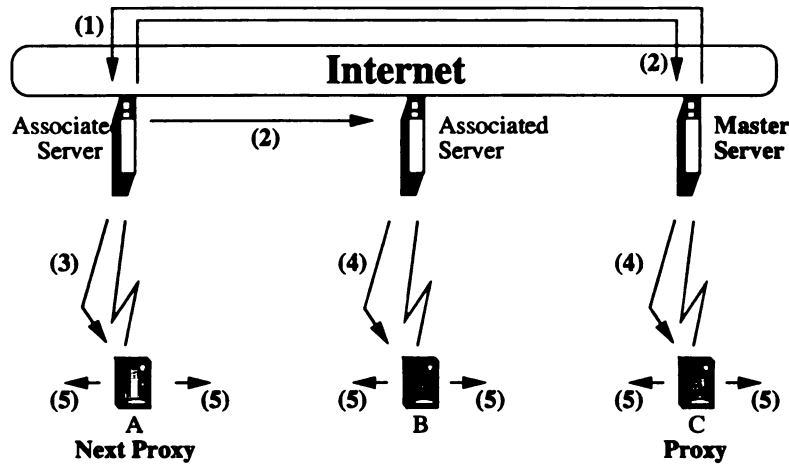
4.3.3 Proxy scheduling

One of the attractive advantages of the *CHUM*₂ network is to save telecommunication cost for downloading multimedia data from the Internet. This benefit comes from the sharing of proxy role among other participating peers in the network. Only one proxy pays for the telecommunication connection at the moment and other peers share the connection with no charge. It is desirable for all participating peers to evenly share the total cost but it is difficult to share the cost fairly because of the dynamic membership change of peers. In spite of the dynamicity, the proxy scheduling should be as fair as possible.

In an one-tier network, the proxy maintains all peer information and selects the next proxy as well as updates a newly introduced peer. In a two-tier network, the

master server performs the proxy scheduling because it maintains the membership information of peers. When a proxy serves a $CHUM_2$ network for a fixed amount of time and there is at least one peer in the network, the proxy role will be moved to another peer. The master server maintains a circular scheduling list of peer members and performs a round robin scheduling. Other methods for fair sharing of the proxy role are possible, but we do not discuss them here, but rather refer to [100]. When a new peer is introduced, the peer will be placed right after the proxy in the circular scheduling list. The reason is because a new member should serve soon as a proxy to ensure that they appropriately participate before benefiting from the service of other peers.

Fig. 4.6 displays the steps for the transition of a proxy role to another peer and the action that follows. The current master server sends a notice packet to the associated server of the next scheduled proxy when the current proxy has finished serving a given amount of time. The packet contains the current peer membership information and the global topology of the wireless $CHUM_2$ network. Upon receiving the information, the associated server notifies all other associated servers by transmitting its transport information such as its IP and port number. In addition, the would-be master server determines the new GID and group key for the new proxy, and distributes the group information to all other associated servers. The associated server passes the new group information and directs its peer to be the next proxy. Then, the associated server itself plays a role of the master server. The other associated servers inform their associated peers about the transport information of the new



- (1) Master server notifies next proxy role to its associated server
- (2) Associated server reports proxy transition to other servers
- (3) Associated server tells its peer to be the next proxy
- (4) Other chumcast server informs the new master server information
- (5) All peers transmit periodic beacon with new master server info

Figure 4.6: The sequence of proxy role transition in a $CHUM_2$ network

master server as well as the new group information. Any peer receiving the new IP and port number updates the master server information and uses the new values for its periodic service beacon. The NID will not be changed. The newly selected proxy includes a new GID in any packet transmitted and encrypts all data packets with a new symmetric group key.

4.3.4 Network partition management

It is possible that a $CHUM_2$ network is partitioned into two or more subnetworks because some forwarding peers between the proxy and the other peers move to another place, and data packets from the proxy cannot be delivered to the peers in the partitioned subnetwork(s). The isolated peers may encounter gaps in their mul-

timedia data unless they have enough buffered data before the reestablishment of the connection with the proxy. It is not always expected that some peers may fill the broken connection and become a bridge between the partitioned network and the proxy. Rather, it is reasonable for the master server to elect a new proxy among peers in the partitioned network and to resume service within the partitioned network.

In one-tier networks, the network partition is detected by the peers that experience gaps in their streaming data. However, in two-tier networks, the master server has responsibility to detect a partition in a $CHUM_2$ network because it maintains the global topology of a $CHUM_2$ network. The server may easily detect a network partition by applying a transitive closure operation on the peer connectivity boolean matrix maintained by the master server. If the proxy may reach all peers in the resultant transitive closure connectivity matrix, there is no partition in the network. If, however, there are some peers that the proxy cannot access, the wireless $CHUM_2$ network is partitioned. The master server selects the peer(s) in the partitioned network and notifies their associated servers about the isolation from the proxy. In addition, the master server assigns one peer as a proxy in the partitioned network. The associated server of the selected proxy generates new NID, GID and group key, and distributes them to other associated servers whose peers are in the partitioned network. The master server of the original $CHUM_2$ network deletes the peer information of the partitioned network from both the membership information and the neighbor connectivity information. The partitioned network may resume its

operation as the group information is shared by all peers, and the newly elected proxy starts downloading the same content from the Internet.

4.4 Security

Wireless *ad hoc* networks are inherently insecure because each host works as a router that may function maliciously. Moreover, the transmission mechanism is broadcast, which is open to nearby hosts. When a packet is not encrypted, eavesdropping is possible in *ad hoc* networks. Much research has focused on secure routing protocols for *ad hoc* networks [121, 122]. Secure routing protocols are needed when some attacking hosts inject malicious packets causing routing loops, routing blackholes, and unnecessary consumption of valuable network resources such as bandwidth or computation power. This section describes the security mechanisms in CHUM networks and the requirements for secure data sharing.

In a $CHUM_2$ network, data packets are rebroadcasted by a set of peers that is selected by the master server. Any peer that is not in the set will receive data packets, but not rebroadcast them. If an eavesdropping mobile host is near a peer that is not in the rebroadcasting set, the host has no chance to receive the packets. Suppose, the eavesdropping host is near a peer that is one of the rebroadcasting peers. The host may hear the packets, but it cannot understand the content of the packets because all packets are encrypted with a symmetric group key that is shared only by member peers of the $CHUM_2$ network. The group key is generated by an master server and passed to the associated servers that deliver the key to their associated

peers through 3G connections. The eavesdropping host is unable to capture the group key unless it has a cooperating server associated with itself. However, it is not allowed for an unauthorized server to be involved in a $CHUM_2$ network, because each server is publicly known to other servers and it must prove itself with a signature.

One way for a freeriding host to receive a symmetric group key comes from the help of a CHUM server on the Internet. The freeriding host may normally contact one of the servers, which may communicate with the current master server in order to receive the group ID and group key as well as to register its associated peer that will be dishonest in the $CHUM_2$ network. This scenario also fails because the freeriding host should be the next proxy scheduled and the host would not serve the $CHUM_2$ network. As a result, this case may result in a blackout of data to all peers in the network for a while. The host becomes a passive attacker rather than an eavesdropper to the network by not serving as a proxy. Sometimes, it is a case that current proxy may leave or fail during its service period. Both passive attack and inevitable failure cases may cause some levels of blackouts in the peers, and the blackout effect may be reduced by buffering data in advance for failure recovery. The master server may maintain a blacklist of the passive attackers' ID such as MAC address in order to reduce the chance of blackouts from the passive attacks.

Each peer keeps a data timer for detecting periodic data packet arrivals. When a peer senses that its data timer has expired, it notifies the silence to its associated server. The associated servers receiving the notification tell the current master server so that the master server corrects the problem. The proxy sends heartbeats to the

master server, more frequently than that as a peer to upload its neighborhood information to its associated server. If no heartbeat is received within a period of time, the master server assumes that the current proxy has left or failed. Then it chooses a new proxy and performs the handover procedure.

A $CHUM_2$ network does not worry about the routing-related attacking problems such as routing loops, routing blackholes, routing grayholes, etc. In an *ad hoc* network, each host works as a router and a host's attack to the network may cause several routing problems. They come from the assumption that all participating hosts trust each other about the routing information generated in the network. False routing information from an attacking host results in the packets that are forwarded but not delivered to their destination, or delivered only to a specific host. In a $CHUM_2$ network, however, no routing information is exchanged among peers. Moreover, not all peers are needed to rebroadcast data packets, but some selected ones retransmit data packets after a random delay from packet arrival. An attacking host may merely send jamming signals over its one-hop transmission range to disturb a $CHUM_2$ network. There will be no further investigation of the physical layer denial of service attacks such as jamming.

4.5 Buffer management in peers

Video traffic is often bursty due to the encoding scheme and the content variation between video scenes. Smoothing by the content provider may reduce the burstiness of the data traffic. Research suggests that unicast with multiple hops and multicast



smoothing by the source reduce both the peak and variability of the bandwidth requirements if buffer spaces are placed at gateway routers or network proxies for the data stream [123, 124]. Peers in a $CHUM_2$ network perform as proxies and packet forwarders for some time, and the buffer space in peers may reduce the burstiness of multimedia traffic. In addition, the buffer space in peers enables continuous playback of multimedia data if there are gaps in transmissions to the peers. The buffer size of peers is related to the timing mechanism in a $CHUM_2$ network, and this section discusses the relationship between the buffer size and the timing issues. Three timing related periods are introduced and the minimum buffer size may be specified from one of the three timing periods.

As previously stated, every peer in a $CHUM_2$ network discovers its neighbors by HELLO packets and delivers the neighborhood information to its associated server periodically. The information is transmitted to the master server that maintains an up-to-date $CHUM_2$ network topology. Each associated server maintains a neighborhood notification timer, which is expired when its associated peer does not upload its neighborhood information within a time period of T_N . When an associated server detects that its neighborhood timer has expired, it assumes that its associated peer has left the $CHUM_2$ network and notifies the missing peer to the master server. The master server performs the $CHUM_2$ network management tasks described in section 4.3, and the associated server resets its timer. When the neighborhood information is updated, its timer is reset.

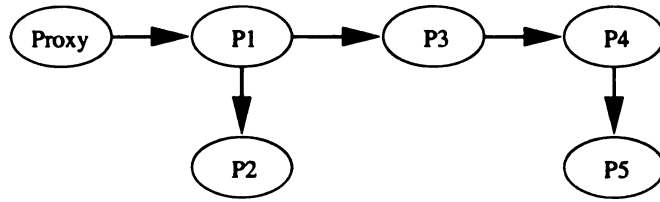


Figure 4.7: A tree of data delivery from a proxy to all peers with some rebroadcasting peers

Each peer also maintains a data timer for data packet arrivals. When no packets are received within a period of T_D , the data timer expires. There are two reasons that a peer does not receive data packets from its neighbor rebroadcasting peer or from the proxy. First, the peer's neighbor rebroadcasting peer has moved away or has failed. Second, the neighbor rebroadcasting peer is selfish and do not rebroadcast the packets. The selfish peer may be detected thanks to the data timer and the promiscuous mode between peers. In Fig. 4.7, the proxy sends data packets, and peers P1, P3, and P4 rebroadcast the packets. Suppose P3 is selfish, and receives packets from P1 but does not rebroadcast them. Then, the data timers in P4 and P5 will expire and their associated servers are informed about the missing data. On receiving the reports from the associated servers, the master server decides which peer is not cooperating. If the neighborhood timer of the associated server for P3 has not expired, then P3 becomes a suspect. All rebroadcasting peers store the timestamp of the last packet received promiscuously, and the timestamp may be known to the master server. If P1 does not receive packets from P3 promiscuously, P3 is determined as being selfish. However, if P1 keeps detecting P3's packets, then P4 is determined to be selfish.

When a proxy finishes its service time, the proxy role is transferred to the scheduled next peer. The current master server delivers all $CHUM_2$ network information to the next master server, including all peer information, logical $CHUM_2$ network connection information, and $CHUM_2$ network management information such as blacklists. Suppose T_{trans} is the time to transmit the $CHUM_2$ network information from the current master server to the next master server. T_{trans} varies depending on the size of the total information as well as the wired network condition such as the round trip time. The next master server is able to estimate T_{trans} with the previous history of T_{trans} and the current measurement of the current transmission time, which operates similarly to the computation of the smoothed round-trip time estimate in TCP.

The minimum buffer size in peers B may be computed as follows:

$$B = T * R * F \quad (4.1)$$

where $T = \text{Max}(T_{trans}, T_N, T_D)$, R is a peer's playback rate of frames per second, and F is a frame size. The value R is a constant that is given by the content provider. All frame sizes F are the same in CBR traffic, but may be different in VBR. To estimate the value F for VBR traffic, a proxy measures the average frame size during its service time and reports it to the master server, which estimates the new value F with the current average size and the history of the previous value. The current master server computes the minimum buffer size B and delivers it to the next master server while the proxy role is transferred. The new master server notifies the suggested minimum

1

value B to all associated servers that direct their associated peers to prepare the minimum buffer space for the multimedia data.

4.6 Cost savings in a $CHUM_2$ network

One of the most attractive features of the $CHUM_2$ network is the saving of the telecommunication cost of downloaded multimedia data from the Internet. In a $CHUM_2$ network with two or more participating peers, all peers receive the benefit of cost reduction by sharing the outside WWAN connection of the proxy. Even in a small-sized $CHUM_2$ network, a joining peer receives large cost savings as well as provides benefits to other participating peers. However, as the $CHUM_2$ network grows larger, the received benefits of the existing peers are already enough and an additional joining peer contributes only a marginal increase of saving to the peers. However, the joining peer receives similar benefits as that of the existing peers. This section shows the amount of cost saving as the network size (the number of peers in the network) increases, and the relationship between the number of peers and the amount of cost saving.

The simulation of the $CHUM_2$ network involves up to 20 peers that download multimedia data from the Internet for 30 minutes. According to [125], each frame size ranges from 1K byte to 30K byte depending on the video contents and their variations. Because $CHUM_2$ network traffic is targetted to much smaller LCD screens such as PDAs and mobile phones, the simulation generates frames sized from 512 byte to 4K byte. The playback rate varies from 2 frames per second to 16 frames per second.

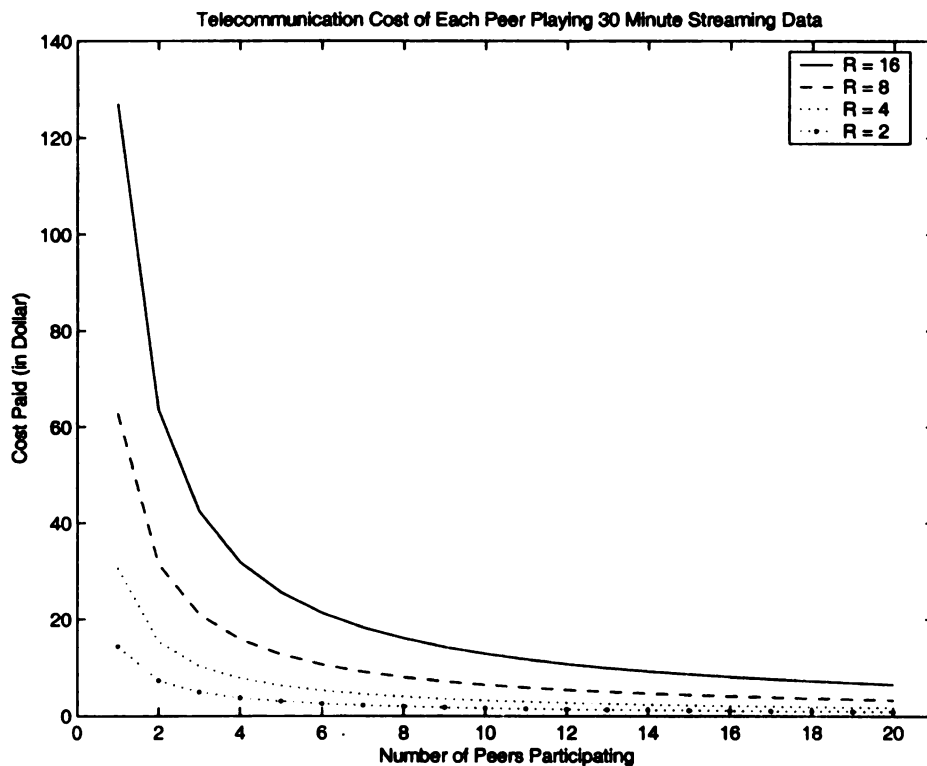


Figure 4.8: Cost of Each Peer Playing 30 min Streaming Data

Each peer serves as a proxy for 30 seconds and the scheduled next proxy takes the role. When a peer joins the network, it is assigned as a next scheduled proxy. The round robin scheduling is used throughout the simulation. The simulation runs with the neighborhood timeout (T_N) value of 10 seconds. That is, for every 10 seconds, each peer uploads the neighbor information to its associated server. All participating peers are honest and their data timers will not expire. This simulation adopts 512 byte per 0.1 cent, which is the rate of the CDMA2000 1xEV-DO 3G service for streaming data in Korea. Both the uploaded neighbor information and the downloaded group information are packed in the packet size of 512 byte.

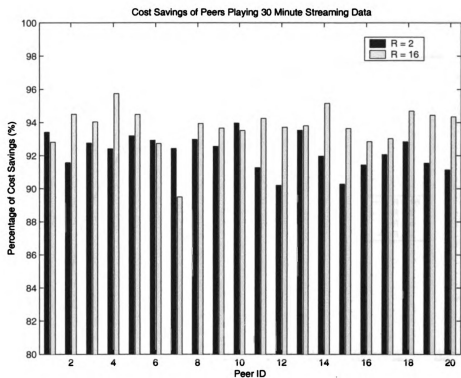


Figure 4.9: Cost Savings of Each Peer with 20 Participating Peers

Fig. 4.8 shows the telecommunication cost for 30 minute (the normal length of a situation comedy or news) streaming data downloaded from a live TV content provider with varying number of peers in a $CHUM_2$ network. As the playback rate R increases, the quality of the video may increase. However, a single user alone may not afford to enjoy the streaming data because of the high telecommunication cost paid. As the number of peers increases in a $CHUM_2$ network, the cost paid by each peer drops dramatically. The current charge plan seems to be expensive, but it is expected that the cost to access the Internet may decrease in near future.

Fig. 4.9 displays the percentage of cost savings when 20 peers participate in a $CHUM_2$ network. One peer forms a $CHUM_2$ network at zero second and

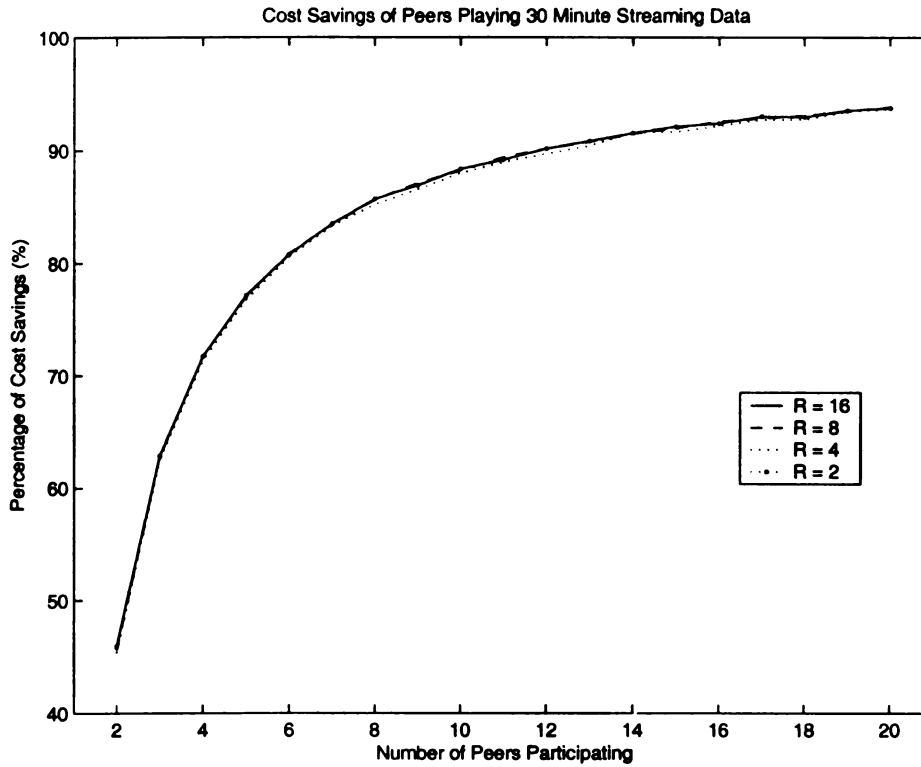


Figure 4.10: Average Cost Savings of Peers with Varying Number of Peers

the subsequent peer joins after the interval from the previous join event, which is exponentially distributed with an average of 20 seconds. After 25 minutes, peers start to leave at a time exponentially distributed with an average of 15 seconds from the previous leave. Fig. 4.9 illustrates the special case of cost saving with 20 peers. The 20th peer joins at the 344 second and the 19th peer leaves at the 1785 seconds. At least one peer stays throughout the simulation. With 20 peers, the cost saving varies from 90% to 93% when $R = 2$ and from 89% to 96% when $R = 16$.

Fig. 4.10 averages the percentage of the cost saving with a different number of cooperating peers from 2 to 20. With the network size of 2, the cost saving is not 50% because of the management packet transmissions between peers and their associated

servers. As the number of peers increase, the percentage of the cost saving increases as well. The percentage reaches 80% with 6 peers when $R = 16, 8$, and 4 , and with 7 peers when $R = 2$. With 20 peers, the cost savings reach at 94% regardless of the playback rate R . It is expected that the telecommunication cost for data traffic will reduce as the technology advances and the market expands. Moreover, the small size of the $CHUM_2$ network may provide additional benefits to the wireless multimedia users as well as provide catalysts to the related industries.

4.7 Summary

This chapter proposes an approach for mobile peers to reduce the telecommunication cost by sharing the connection from one of the peers, called the *proxy*. All peers take turns to become a proxy, which downloads multimedia data from the Internet and chumcasts the data to its peers. Each peer in a wireless *ad hoc* network is associated with a server located in wired network. The master server manages peer information, selects rebroadcasting peers, schedules the next proxy, and detects network partitioning. By uploading all neighborhood information periodically to associated servers and then to the master server, all peers are free from performing those tasks. The reduction of the tasks by the peers saves wireless bandwidth. By cooperating with wired and wireless networks, the $CHUM_2$ network operates in a more secure environment than if it operates only in a pure *ad hoc* network. In addition, the peers save a large amount of the telecommunication cost even if a small $CHUM_2$ network is formed.

Chapter 5

Parallel Downloading of CHUM Networks

5.1 Introduction

In the previous two chapters, peers form a $CHUM_1$ network or $CHUM_2$ network to download multimedia streaming data from the Internet. This chapter presents a different CHUM network, labeled $CHUM_p$ network, in which several mobile devices, called peers, cooperate to download their assigned portion of the content using a 3G connection in order to reduce the overall cost to download the content. Then, the peers exchange their assigned portion of the content with other peers via a wireless *ad hoc* network and the peers reconstruct the content. Fig. 1.4 in chapter 1 illustrates a situation in which several mobile users download a popular file from a content provider (CP). Suppose users of the mobile devices wish to download a mobile game

program in order to play interactively with each other. Fig. 1.5 and Fig. 1.6 display the mechanism for partitioning the target file and downloading the assigned portions by the participating peers. Each peer is associated with a server located within the Internet. The associated server cooperate to schedule each peer to download its unique portion of the content. As the peers complete downloading the specified portion of the file, they use their *ad hoc* connection to exchange the content with other peers in order to reconstruct the target file. The master server schedules the distribution pattern for exchanging content between the peers. A controlled content distribution algorithm is required because each peer will send/receive data to/from all other peers, and many packet collisions are expected when the peers transmit their packets in an uncontrolled manner.

This chapter focuses mainly on the two data distribution methods among participating peers. Both methods do not use any *ad hoc* routing algorithm, but all peers use simple broadcasting of packets to their neighbors. The neighbors decide the next action depending on the distribution method. One method employs a *per-packet* policy [126]. When a peer transmits one unit-sized segment of downloaded data that is new to other peers, the data packet travels over the $CHUM_p$ network by rebroadcasting from some selected peers, if needed. If all peers receive the packet, the scheduled next peer transmits another unit-sized segment of downloaded data. Per-packet transmission repeats until all peers receive the complete content. The other method is based on a *per-peer* policy. All peers are ordered to transmit their packets one peer at a time in a circular list. They all know which peer is its predecessor in the

list. When a peer transmits some number of data packets, the content is delivered only to the sender's one-hop neighbors. At the end of the current peer's transmission, the peer sends a special packet, called the DONE packet, that triggers the transmission of the next peer in the list. The circular transmission process finishes when all peers receive the complete target file. The reason to control the transmission sequence among peers is to avoid large numbers of packet collisions, called the broadcast storm problem [61].

5.2 Network operation

5.2.1 Formation of a $CHUM_p$ network

The $CHUM_p$ network in this chapter focuses on the sharing scheme in which all peers download partial content in parallel with their own 3G connections, and exchange them with other peers using their shared cost-free *ad hoc* connection until all peers receive the complete content. Suppose some friends intend to download an interesting mobile game program that they want to store on their mobile devices and play together interactively. One of the mobile device users initiates the process of $CHUM_p$ network formation. The device user makes a connection to its ISP to access the Internet and contacts one of the CHUM servers on the Internet. The CHUM network formation is completely dependent on the request of the initiating mobile device. The device may want to construct a $CHUM_1$ network or a $CHUM_2$ network for downloading streaming data, or a $CHUM_p$ network for non-streaming data.

The CHUM servers help to build an appropriate CHUM network according to the device user's need. The device gives information about its $CHUM_p$ network such as the minimum number of participating mobile devices and the file specification to be downloaded. The server becomes the associated server of the device and accesses the CP in order to fetch detailed information such as the size of the file. In addition, the server generates a group ID (GID), group key, a $CHUM_p$ network ID (NID), and a unique peer ID (PID), which is similar to the formation of the $CHUM_2$ network. The usage of the information is already described in section 4.2.

The server of the initiating mobile device plays two roles in the $CHUM_p$ network: the master server and the associated server. The master server mainly deals with several management tasks for the $CHUM_p$ network. First, it creates and maintains group information and assigns new PID when new device joins the network. Second, it manages the membership and the logical topology of the $CHUM_p$ network. Third, the master server schedules the downloading process which device to download which portion of the target file. Fourth, the master server detects and recovers from several failures, if exist, such as peer missing and network partitions. The associated server contacts its associated mobile peer and collects local neighborhood information from its peer. In addition, the associated server selects a set of rebroadcasting peers for its associated peer when the peer broadcasts downloaded data to other peers. The $CHUM_p$ network includes one master server and each peer has its associated server. An associated server may serve several peers at the same time.

The remaining cooperating devices hear the beacon and send a JOIN packet to the sender of the beacon. The process of joining an existing $CHUM_p$ network is similar to that of $CHUM_2$ network, which is described in section 4.2. As peers join the $CHUM_p$ network, the master server compares the current number of $CHUM_p$ members and the minimum number of participating peers that was specified by the initiating peer. The minimum number will be at least two since there is no advantage to form a $CHUM_p$ network with only one member. When the number of peers that have joined the network reaches the minimum number, the master server computes the download schedule for all participating peers. The master server transmits the related download schedule to each associated server so that all peers can download their specified portion of the target file from the CP.

When a peer joins a $CHUM_p$ network, it transmits a beacon periodically for other mobile devices to join the network. In the middle of the downloading process, any mobile device may join the network. The master server updates the membership and recomputes the download schedule. The master server should consider the fairness of the download cost for all peers in the recomputation of the scheduling. After some time has passed, if the total amount of data downloaded by all peers exceeds a threshold, the master server directs all associated servers that no more beacons are needed from their associated peers, and no more peers will join the $CHUM_p$ network.

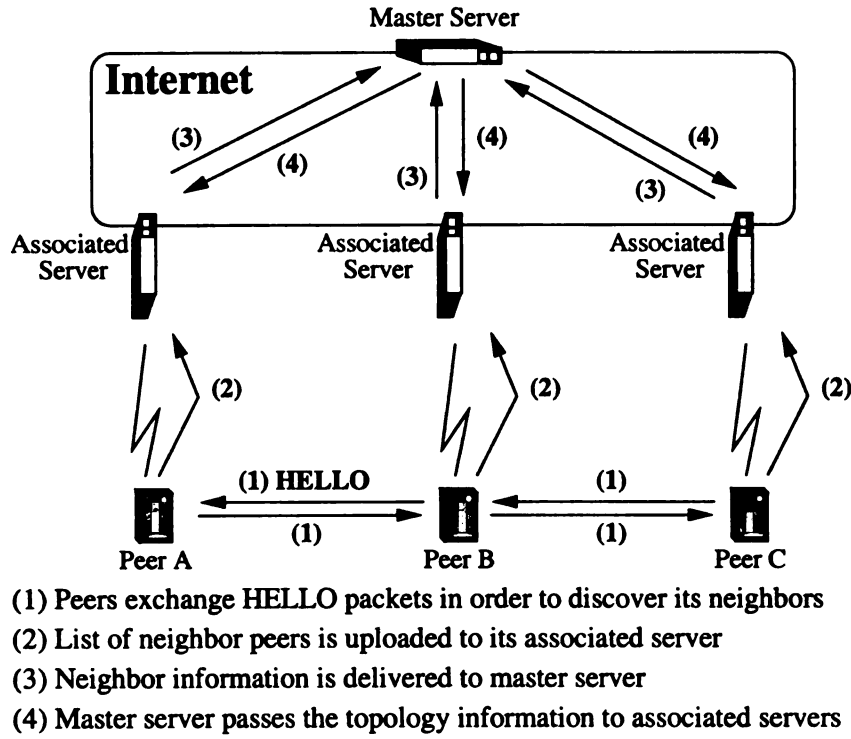


Figure 5.1: The sequence of processing neighbor information

5.2.2 Membership management

When a peer joins a $CHUM_p$ network, described in the previous section, the master server updates the membership information. In addition, both the master server and the associated server require neighborhood information to perform their tasks. Fig. 5.1 illustrates how the neighborhood information is collected from the peers and used in a $CHUM_p$ network. Each peer detects its one-hop neighbors by exchanging periodic HELLO packets using the *ad hoc* network. The HELLO packet contains the GID and the PID of the sender. When a peer receives a HELLO packet, it examines the GID in the packet. Only the peers with the same GID process HELLO packets as well as keep the neighborhood information. If there is a change, the peer

uploads the neighbor list with other information, such as the download status bit map described in section 5.3.1. The associated server delivers the change to the master server. The master server updates the change of the membership information and the neighborhood information.

The master server is able to create a global topology of the $CHUM_p$ network by synthesizing the partial neighborhood information of each peer. When the master server receives new neighborhood information, it updates the global topology of the network. The global topology is essential for both the master server and associated servers to compute the download schedule and the minimum rebroadcasting set. The master server maintains a boolean connectivity matrix. The matrix represents the global topology of the $CHUM_p$ network. The master server computes the download schedule with the membership information. The master server also distributes the topology information to all associated servers so that they can compute the rebroadcasting set of peers.

5.2.3 Download scheduling

The master server schedules each peer to download the same or similar amounts of the target file. Moreover, when all portions are collected, a file should be constructed that is identical to the target file. Suppose N is the size of the target file to download, and D is the size of assigned portion downloaded by each peer in each round. The value D is equal to $k * u$, where u is a payload of each packet downloaded at a time from the CP and k is an integer. The master server assumes $N = D * t$, where t is

an integer. If the given file size N is not a multiple of D , the master server adjusts the value N by adding an amount by the equation $N' = N + ([N/D] * D - N)$. The value N' becomes a multiple of D and is set to be a new size of the target file. In each round, the total of $D * p$ bytes are downloaded by all peers, where p is the current number of peers. The master server executes the following algorithm to assign the beginning location of the target file for each peer.

```
for (i = 0; i < p; i++)
{ download-peer(i, base+i * D); }
base = base + D * p;
```

Each peer is indexed from zero to $p-1$ by the master server and the peer with index i should download a portion of the target file starting at the second argument of the function *download-peer* with the size of D bytes. The master server notifies all associated servers about the location where to download for their peers. The master server keeps a mapping table that identifies which peer has downloaded which portion of the file. The value *base* is initialized to zero and increased at the end of each round.

In the middle of the downloading process, some mobile devices may want to join the existing $CHUM_p$ network. In this case, the master server needs to decide whether to accept the new devices. The master server computes the minimum value of n that satisfies the following inequality

$$D * p_r + D * n * (r + 1) \leq N'' \quad (5.1)$$

where r is the current round of download completed, p_r is the number of participating peers in round r , $n = p_{r+1} - p_r$, a number of newly joining peers, and $N'' = N' - base$, the remaining bytes to download. That is, all existing peers will download D bytes in the $(r + 1)^{th}$ round, and the newly joining n peers will download $(r + 1) * D$ bytes. To be fair, all participating peers (including new peers) download the same or similar amounts of data at the end of downloading process. If the minimum value n is equal to zero, no more new devices will be accepted. Otherwise, the first n peers registered are accepted as peers to participate. At the end of each round, the master server checks whether there exists a positive integer n that satisfies equation 5.1. If there is no such value n , there is no need to add more mobile devices. The master server directs all associated servers that their peers should stop transmitting beacons.

At the final round, the remaining bytes may be smaller than $D * p$. The master server finds the smallest index rem such that $D * rem$ becomes larger than the remaining bytes to download. Then, the master server assigns the remaining portions of the file to the first rem peers, or randomly selects rem number of peers. After the completion of the final round, the complete target file will be stored at the peers in the *ad hoc CHUM_p* network.

5.3 Per-packet based content distribution

5.3.1 Content distribution scheduling

In order to reconstruct a file, every peer in the network is a sender as well as a receiver. If all peers, as senders, transmit their data to the $CHUM_p$ network in an uncontrolled manner, the broadcast storm problem [61] may arise. Each peer needs a controlled way of broadcasting its received data to other member peers, and some of them need to rebroadcast the received packets if some peers are out of transmission range of another.

Selection of rebroadcasting peers

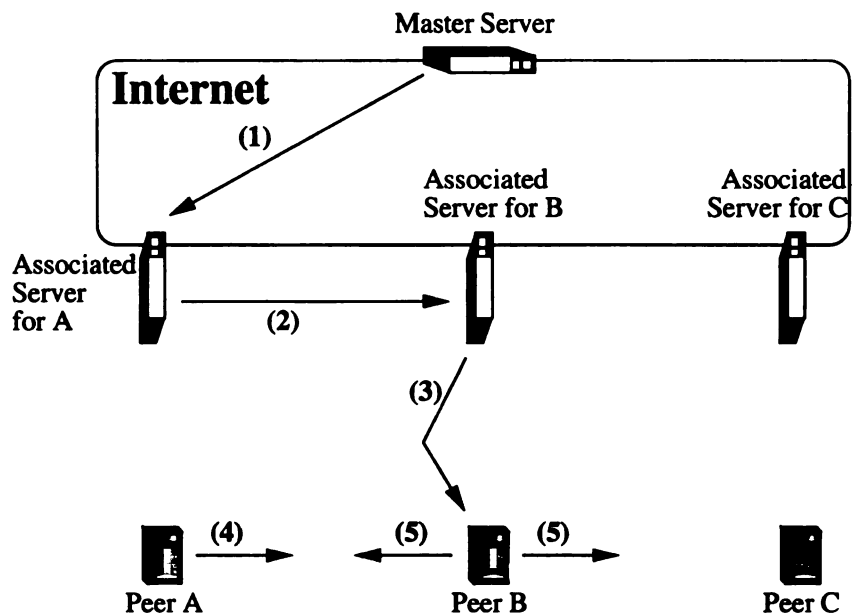
Much research have proposed distributed algorithms for selecting the rebroadcasting peers while using local neighbor information [62, 66, 67, 119]. The research proposes two-hop neighbor knowledge to minimize the number of rebroadcasting peers. Because wireless broadcasting may cause the duplication of packets and the contention of the wireless medium, it is important to select carefully the set of minimum rebroadcasting peers in the $CHUM_p$ network.

The problem of selecting the minimum rebroadcasting set is very similar to the MCDS (Minimum Connected Dominating Set) problem. The authors in [62] mentioned that the minimum rebroadcasting set problem is harder than the MCDS problem, which has been proved to be NP-complete. Several wireless broadcasting algorithms [62, 66, 67, 119] have been proposed for the approximation of MCDS to

compute the backbone wireless peers or the rebroadcasting peers using local neighborhood information. Starting from the source peer, two-hop knowledge helps to decide the next rebroadcasting peers in a completely distributed manner.

Wireless *ad hoc* broadcasting algorithms seldom use the global topology of a network due to the lack of a centralized controlling peer and the scarcity of wireless bandwidth for exchanging neighborhood information with other hosts throughout the network. The servers in a $CHUM_p$ network utilizes the global topology of the $CHUM_p$ wireless network.

The best known algorithm for the MCDS problem is the Berman's algorithm [120], which uses a global topology. The algorithm generates a set of connected nodes and then merges the set into one connected dominating set. The Berman's algorithm requires $O(D)$ steps and an approximation ratio of $\ln \Delta + 3$, where D is the diameter of the network and Δ is the maximum degree of the tree. In the $CHUM_p$ network, the local neighborhood information from each peer provides a hint to generate a mesh of connected peers. The master server merges the connected peers into one connected set of peers, which represents the global topology of the $CHUM_p$ network. The master server passes the information to all associated servers. Each associated server produces the minimum set of rebroadcasting peers for its associated peer working as a source peer. As the source peer changes, so does the rebroadcasting set for the peer. Each associated server notifies the involvement of the rebroadcasting to other associated servers of the peers in the set. Each responsible associated server directs its peer(s) to rebroadcast packets when it receives data packets originated from the



- (1) Master server passes global topology of CHUM network
- (2) Associated server computes and passes rebroadcasting set for A
- (3) Involved associated server notifies to its peer to rebroadcast
- (4) Source peer transmits its downloaded data to its neighbors
- (5) Peers in the set rebroadcast received packets originated from A

Figure 5.2: The sequence of generating rebroadcast peer set

specified peer. Fig. 5.2 illustrates the computation and the notification sequence of the rebroadcasting set. In the figure, the associated server for peer A selects peer B as its rebroadcasting set. When peer B receives packets that originated from peer A, it rebroadcasts the packets for peers, such as C, that are located far from peer A.

Data distribution

Fig. 5.3 shows a typical example of the *ad hoc* $CHUM_p$ network topology. Nine peers, labeled A to I, form a $CHUM_p$ network. They have downloaded some portion of data from the Internet. The master server defines the transmission sequence of peers

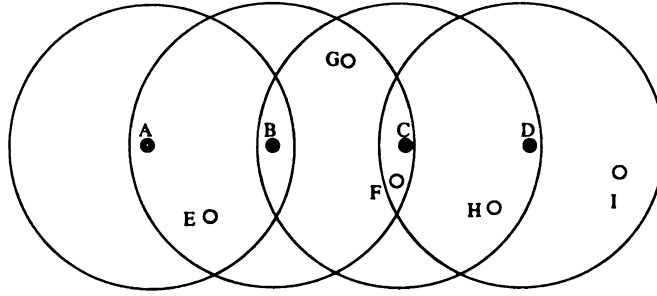


Figure 5.3: A case of an ad hoc network topology

to distribute their downloaded data. Fig. 5.3 assumes the transmission sequence as an alphabetical order of peers from A to I. Every peer knows which peer is its predecessor that triggers the next transmission. As an example, when peer A finishes its data delivery, peer B resumes its transmission. Every peer in the set has a list of rebroadcasting information that describes when to rebroadcast if received packets have originated from a specific peer. For example, the associated server of peer A selects peer B, C, and D as A's rebroadcasting set in Fig. 5.3. In a similar way, peer B's rebroadcasting set will be A, C and D. Peer C has rebroadcasting information that directs to rebroadcast if it receives packets originated from peer A or B. Peer D also has additional knowledge that peer D is the *terminal peer* of the rebroadcasting set. That is, when peer D rebroadcasts, there are no more peers to rebroadcast the same packet from peer D. The terminal peers play an important role to trigger the next sequence from the source peer to transmit its downloaded portion of data.

The $CHUM_p$ network adopts an asynchronous transmission mechanism in which a packet from the predecessor triggers the transmission of the successor's packets. In Fig. 5.3, peer A is the first peer to transmit its data in the sequence. When

peer A broadcasts its data, peer B, C, and D, in that order rebroadcast the packet originated from peer A, which enables all peers to have the content downloaded by peer A. Peer B needs to know when to broadcast its downloaded data. Suppose peer B rebroadcasts its data right after receiving data from its predecessor peer (peer A in this case). This may cause collisions in some peers and data lost because peer B may transmit its downloaded data while peer C may rebroadcast A's packet, which causes peer G to experience a collision. In order to reduce the possibility of collisions, peer B should delay its transmission until peer A's packet is delivered to all peers. The terminal peers are useful to decrease the probability of collisions. When peer A is a source, peer D becomes a terminal peer. In contrast, if peer B is a source, then both peer A and D become terminal peers. When a terminal peer rebroadcasts a data packet, it immediately broadcasts a DONE packet that notices the completion of rebroadcasting in one end of the data flow. The DONE packet propagates backward to the source peer until the next peer in the transmission sequence receives the DONE packet. The packet triggers the broadcast of B's downloaded data. Fig. 5.4 shows the snapshot of data propagation from peer A, B, C and D in the $CHUM_p$ network displayed in Fig. 5.3.

Recovery from relocated rebroadcasting peer

Some peers may move and change their locations during data exchange. In addition, some peers may fail due to the lack of battery power. If some peers in the rebroadcasting set fail or change their locations, the data distribution process may

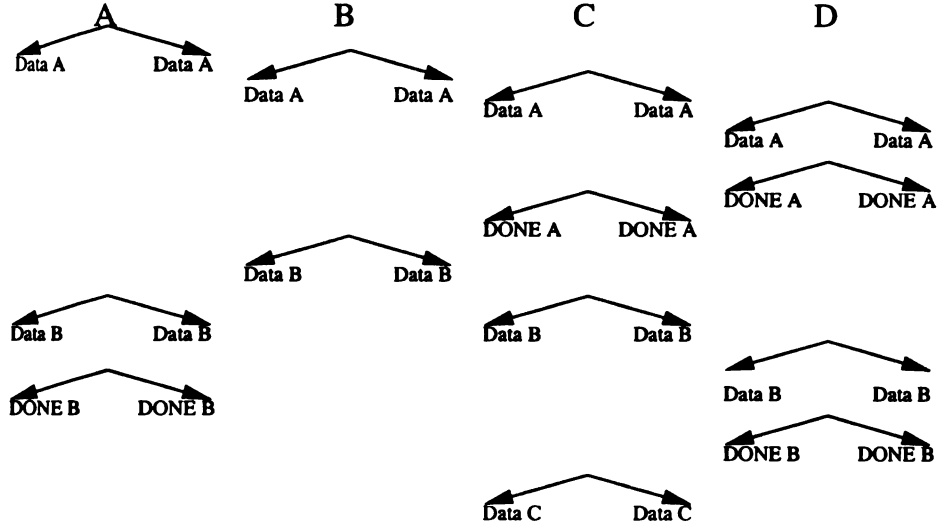


Figure 5.4: A triggering example of Data and DONE packet transmission

stop functioning. Suppose peer C moves toward peer B in Fig. 5.3, and peer C cannot reach peer D. Then, peer D may not receive any packet and the $CHUM_p$ network may stop working. The master server provides a solution to this problem. The master server may receive updated partial neighborhood information from peers by the HELLO packet that detects peer C's location change. The updated global topology information is sent to all associated servers. The associated servers recompute the rebroadcasting set, if needed, for their peers. When the member peers in the set change, they will be notified of the related rebroadcasting peers. This scheme correctly computes the rebroadcasting set one period after the neighbor information is uploaded.

1

Recovery from missing data

Every peer maintains a receiving status bitmap of size t such as $N' = t * u$ where N' and u are defined in section 5.2.3. As the size of the target file is known, so is the size of the bitmap. Each bit indicates whether the mapped content has been received either from the 3G connection or from the *ad hoc* network. A bit is set if the corresponding unit-sized payload is received.

When the final round of downloading is completed and all packet exchanges within the *ad hoc* connection have finished, each peer uploads its bitmap to its associated server. There are many reasons that some peers have not received some portions and the corresponding bits remain to be set. When one or more peers have failed or have left the network, some part of the target file will be missing. This case requires an additional downloading round for the missing portion of the file.

The master server collects the bitmaps from all peers and applies bitwise-OR operation on all bitmaps. If the resulting bitmap contains at least one bit that remains reset, it indicates that no peer downloaded that portion of the file. It is because one or more peers left or failed during the download phase. The master server counts the number of reset bits, and records the position of the missing portion of the file. Then, the server decides which peer(s) to download the missing portion. The extra downloading schedule is delivered to the associated server of the peers involved. The responsible peers resume downloading the assigned portion of data in order to fill the missing gap. This re-downloading process finishes when the result of the bitwise-OR operation on all bitmaps is set for every bit position.

When the complete content of the target file is in the *ad hoc* network, the remaining task is for all peers to share the complete content. The master server maintains the bitmap matrix of the size p by t , where p is the number of peers and t is the total number of payload units. If the element (i, j) in the bitmap matrix has a value of zero, it implies that the i^{th} peer does not have the j^{th} unit segment of the file. The master server scans each column of the matrix whether the column includes any bit reset. If it exists in the j^{th} column, the master server selects the source peer and a rebroadcasting peer set for the j^{th} unit segment. When peers receive the missing segment of data, they update their bitmaps. After completing the scan of all columns of the bitmap matrix and processing any existing gaps, all peers upload their bitmaps to the master server. The master server finishes the missing data recovery process when it finds that all bitmaps contain all ones, which indicates all peers have stored the target file.

5.3.2 Security issue

The security issue discussed in chapter 4 mainly focuses on the avoidance of freeriding and the protection of $CHUM_2$ networks from passive and active attackers. This subsection describes the authentication of the received content as well as the encryption of the content. A $CHUM_p$ network is a closed network in which only the participating member peers share their downloaded content with each other. However, the transmission mechanism of peers is broadcast that is open to any nearby devices. The $CHUM_p$ network should equip with the encryption feature by which the broadcasted

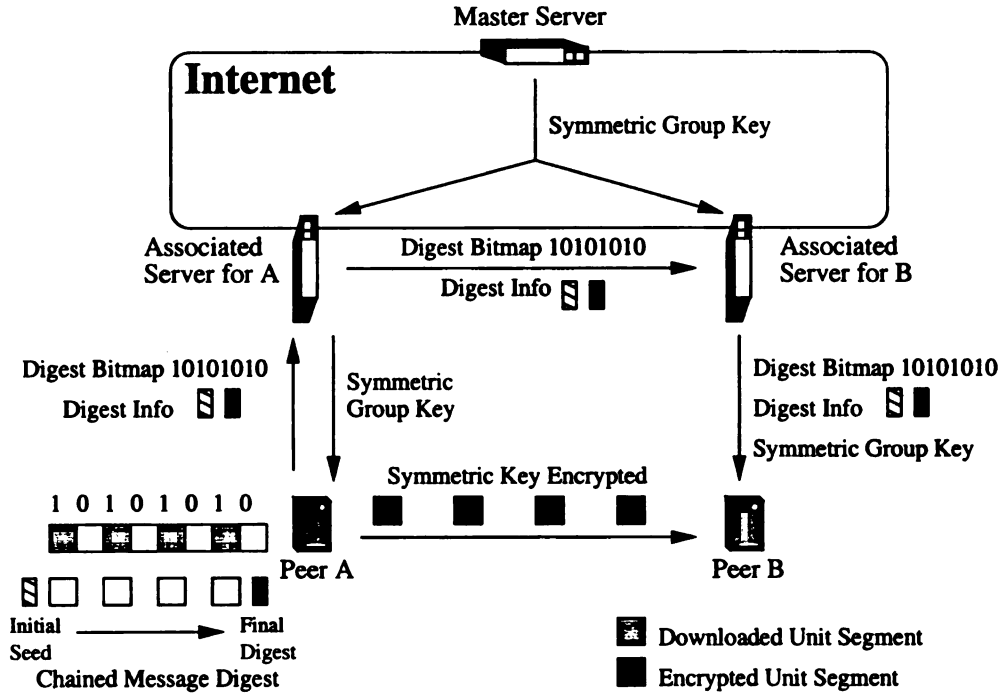


Figure 5.5: $CHUM_p$ network encryption and authentication

content are protected from freeriding devices. Moreover, even though all peers may download a partial content of the target file, they are guaranteed to receive a complete and intact version of the target file. As a result, the security mechanism should contain the authentication feature so that all peers can detect whether the received content from the *ad hoc* connection is altered or contaminated.

Fig. 5.5 displays the security mechanism adopted in the $CHUM_p$ network. In order to prevent the disclosure of downloading content to eavesdropping devices, every source peer encrypts every packet before transmitting to member peers using some popular encryption algorithms such as DES [127] or AES [128]. The receiving peers decrypt the received packets over the *ad hoc* connection with the same symmetric group key as the source peer encrypted. The symmetric group key is created by the

master server when the $CHUM_p$ network is constructed. The master server passes the key to the associated servers using the public key encryption over the wired network. Each server knows the public keys of all other associated servers in the $CHUM_p$ network. Each associated server delivers the symmetric group key to its peer through the peer's 3G connection that is a private communication channel between the server and the peer.

The authentication is necessary because the transmitting packets may be interfered or contaminated by nearby signals. The receiving peer needs to verify that the received content over the *ad hoc* connection is correct. Every peer is responsible for downloading some portion of the target file and transmitting them to other peers via the *ad hoc* link. The sending peer executes a chained message digest algorithm such as MD5 [129] with the creation of an initial seed. Before transmitting the first packet of the unit payload size discussed in section 5.2.3, the peer computes the next seed with both the first unit-sized block and the initial seed. The following block to be transmitted will be the source for computing the next seed. When the last unit block to send is computed, the final digest is the digest for the entire content downloaded and transmitted by the peer.

Every peer maintains a message digest bitmap of the same size of the receiving status bitmap discussed in section 5.3.1. A bit in the digest bitmap is set when a peer computes the digest of the corresponding unit block. When all peers receive the whole file by exchanging the content with other peers, they upload three information to their associated servers: the initial seed, the final digest, and the digest bitmap.

The information will be delivered to each peer via its associated server so that it is able to compute its own digest and compare it with the received one. If both digests mismatch, the receiving status bitmap is NORed with the digest bitmap to nullify the received content from the sending peer of the digest bitmap. The missing content can be recovered described in section 5.3.1. When all peers have both digests that are correctly matched, every peer has the target file in its storage.

5.3.3 Simulation result of per-packet method

This section describes the simulation model and results. The *ns2* network simulator [130] is used. The model assumes each peer downloads portions using its 3G connection that the master server schedules to download, while it exchanges the portions with other peers using its *ad hoc* network. The peers do not experience buffer underrun when they transmit their portions to other peers. The unit packet size of the 3G connection is set to 512 byte. The peers download the file size of 1 MBytes that consists of 2048 unit packets. Each peer has the 802.11 MAC with the transmission range of 250 meters. The number of peers varies from 2 to 10. The peers move at 2 m/s according to the way-point mobility model in a 400 meter by 400 meter grid. Three pause time values are used: 0, 10, and 20 seconds. Each second, each peer uploads its new neighborhood information. In all cases, the *ad hoc* network of the $CHUM_p$ network is connected.

Fig. 5.6 displays the average number of packets used by the peers that may have a fee charged a telecommunication provider. Each value in the figure is the

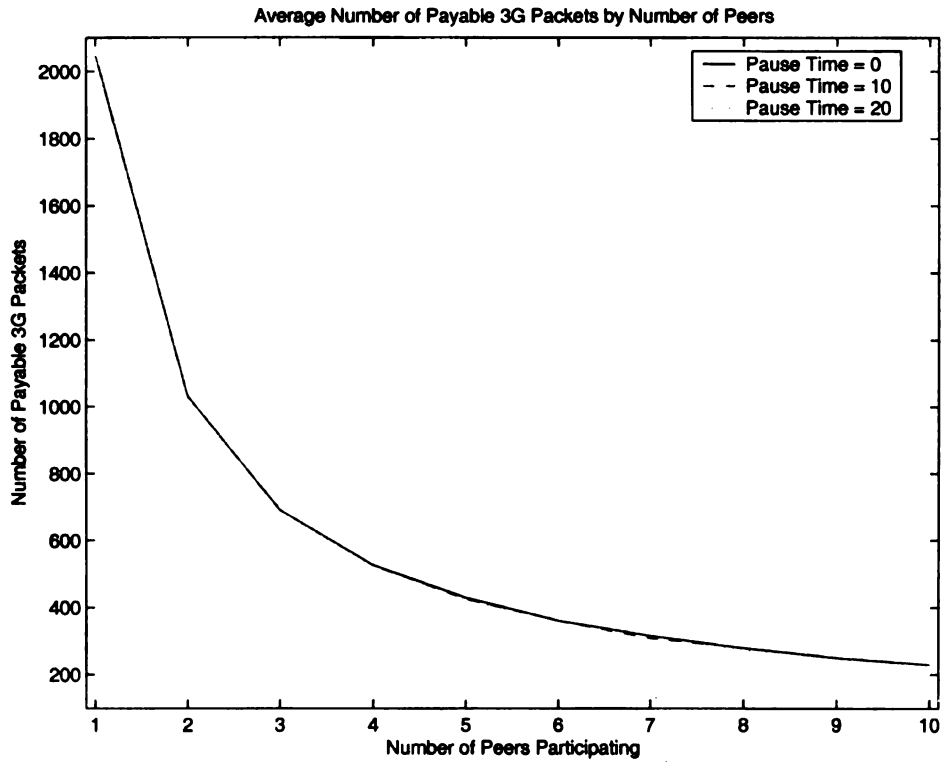


Figure 5.6: The total number of 3G payable packets with different number of peers average of ten runs. As the number of peers increases, the number of fee-based packets decreases substantially. However, an additional peer only reduces a marginal cost when there are already enough peers, whereas the new peer still receives the same benefit of the cost reduction. The pause time does not influence the results due to the short download completion time. The completion time of each simulation is shown in Fig. 5.7. This simulation uses the pause time of zero seconds. The average completion time increases slowly as more peers participate. It is because when the area becomes crowded, the chance of packet collision increases. The $CHUM_p$ network operates with the packet triggering mechanism in which The current packet triggers the next packet transmission. The rebroadcasting peer or the scheduled next source

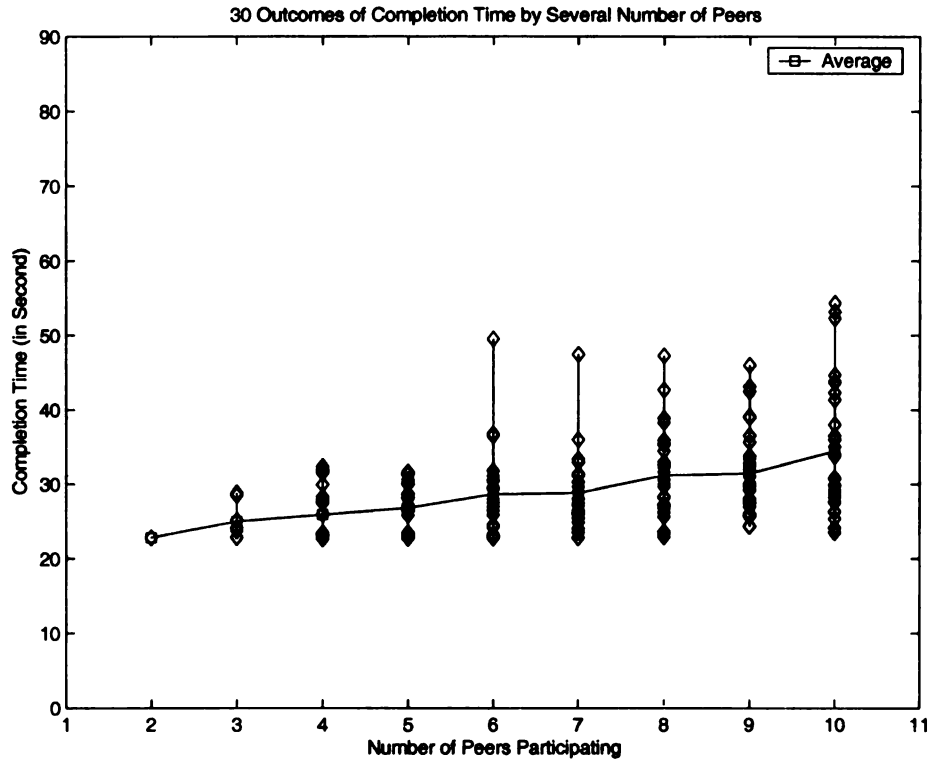


Figure 5.7: Exchange completion time with different number of peers

peer may lose its chance to transmit and waits for receiving the lost packet from its previous transmitting peer.

The $CHUM_p$ network is greatly influenced by the area size. When all peers are located within the transmission range of each other, no collisions are expected because only one peer transmits at a time. Both Fig. 5.8 and Fig 5.9 illustrates how the performance changes due to a change in the grid size. The simulation runs with 10 peers in constant movement. As the grid size increases, so does the completion time and the number of fee-based packets. This is due to the fact that when peers are located in wider area, there needs to be more rebroadcasting peers. Some packets

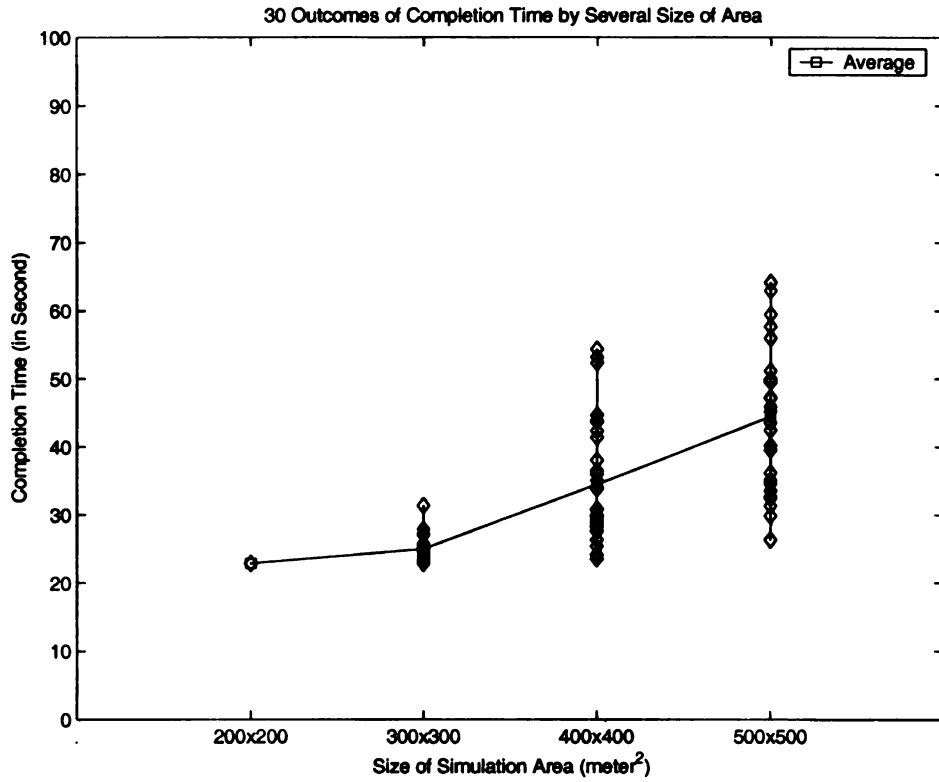


Figure 5.8: Exchange completion time with different simulation area size

may collide and some peer lose the packets, which deprives the chance to rebroadcast the packet or to transmit the next sequenced data packet.

5.4 Per-peer based content distribution

A per-packet based distribution method provides an intuitive understanding of how packets are exchanged. In the distribution method, one packet from one peer is forwarded to all other peers at a time. Then the next scheduled peer takes a turn to transmit its next unit-sized packet after receiving a DONE packet from its predecessor. If peer A in Fig. 5.3 sends two unit-sized packets in a row, it may cause packet collision

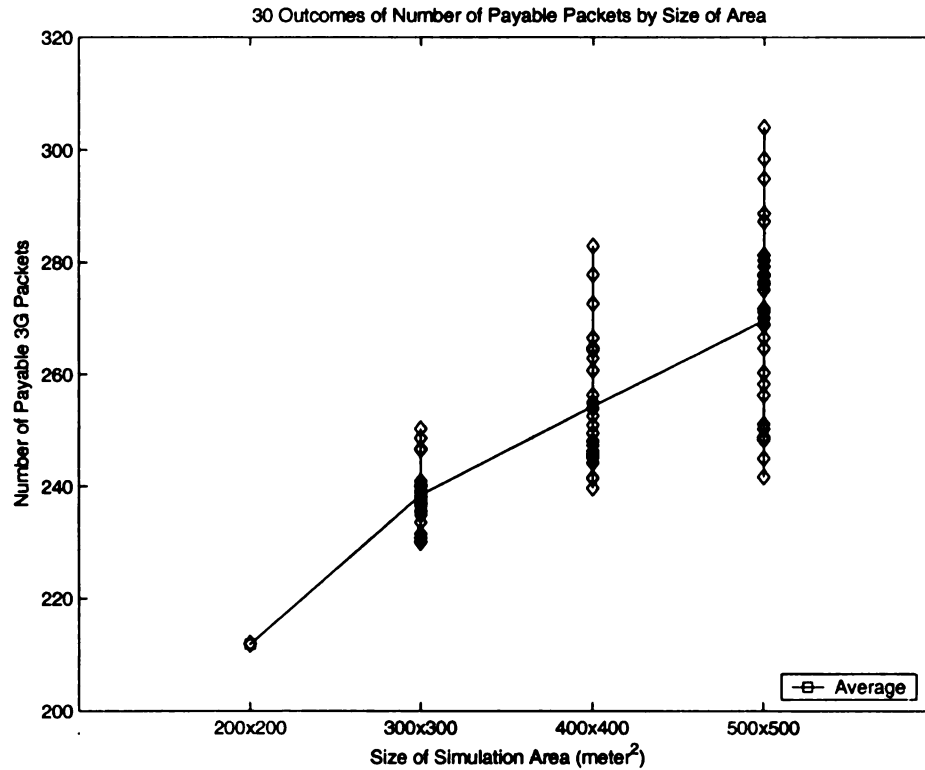


Figure 5.9: The number of 3G payable packets with different simulation area size

because of both peer A's second data packet and B's first rebroadcasting packet. This delivery method, however, shows some inefficiency while transmitting packets. In some topologies including star-shaped networks, peers located at the central area may experience packet collision. The collision causes data retransmission with the aid of bitmap packets during the missing data recovery phase, which uses cost-based telecommunication links. In addition, for some moment, non-data packets such as DONE packets may dominate the *ad hoc* network, which increases the completion time and consumes battery power of the mobile devices.

A per-peer based distribution method decreases the chance of packet collision and increases the time for more data packets transmitted, which results in shorter completion time, less 3G communication costs, and less power consumption than that of a per-packet based method. In the per-peer based distribution, only one peer has a chance to transmit some number of unit-sized data packets to its neighbors at a time. In addition, the data packets are not immediately forwarded, which reduces the possibility of packet collision. Each peer decides which unit-sized data to broadcast depending on the reception status of its neighbors. The neighbors do not rebroadcast any packet immediately, but wait for their transmission turn. When the current transmitting peer has finished broadcasting a given number of packets, it broadcasts a DONE packet. If a scheduled next peer is directly connected with the current peer, the next peer resumes its transmission after selecting which data to send. If the next peer is out of range from the current peer, the DONE packet contains a list of rebroadcasting peers. Any peer finding its PID in the list of the received DONE packet rebroadcasts the packet. The DONE packet is rebroadcasted immediately, if needed.

The master server creates and maintains a transmission sequence as a circular list of peers using global topology information. Fig. 5.10 illustrates one example of a 3-hop $CHUM_p$ network. The master server constructs the list with the following rules. The server prefers the peer that has the largest number of neighbors. Then it selects the peer that is directly connected with the previously selected peer. If there is no unselected peers that have direct connection with the recently selected

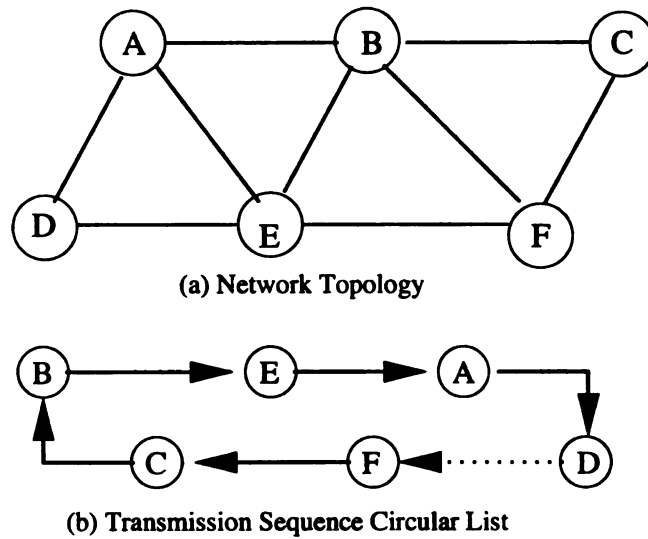


Figure 5.10: A 3-hop network topology and its transmission sequence circular list

peer, the server chooses the peer with the minimum number of hops. Fig. 5.10 (b) is one possible example of the transmission sequence as a circular list. Because peer D and F have no direct connection, the link is shown as a dotted line. Peer D keeps a rebroadcasting set that includes peer E for rebroadcasting a DONE packet. In addition, each peer maintains two-hop neighborhood information. When a peer receives a packet, it immediately knows which peer also receives the same packet. This knowledge may obviate some unnecessary transmissions of data packets.

One important value that each peer maintains for each unit-sized data segment is the number of neighbors that do not have the unit-sized data segment. Because the target file size is known and the data packet size can be defined, e.g., 512 bytes, each peer is able to compute how many unit-sized data are needed to construct the target file. Suppose the target file size is 1 Mbyte and the unit packet size is 512 bytes, then 2048 data packets are needed. Each peer maintains a bitmap size of 2048 bits and

the same-sized integer array called the *benefit* value. The benefit value is zero if all neighbors contain the associated unit-sized data. In case that a peer downloads some portion of the target file from the Internet, the benefit value(s) of the data segment(s) is(are) set to the number of its neighbors. When each peer resumes transmission, it selects the unit-sized data that has the largest benefit value, which increases a chance to discourage unnecessary data transmission by other peers with the aid of the two-hop neighborhood information. In Fig. 5.10, assume peer B sends a data packet whose benefit value is four, and peer A, C, E, and F receive the packet. Peer B's benefit for the data becomes zero. Both peer A and E computes a value of 1 for the data's benefit value, because only peer D does not have the data. Further assume that peer E transmits the data in its turn. Now peer A's benefit value of the data becomes zero and peer A is unnecessary to transmit the data in case of its turn to transmit. If a peer has all zero benefit values in its stored data, it immediately sends DONE packet when it is the peer's turn. This allows other peers with positive benefit values to transmit data.

One improvement on the per-peer based distribution is to provide additional opportunities for data transmission to peers that have substantial data with a positive benefit value. As the data distribution proceeds, the peers in the central area of the network receives more data than that of edge peers. Many edge peers do not have data with a positive benefit value except the data downloaded from the Internet itself. If each peer broadcasts one data packet at a time, several consecutive DONE packets may exchanged in the middle of data distribution, which increases

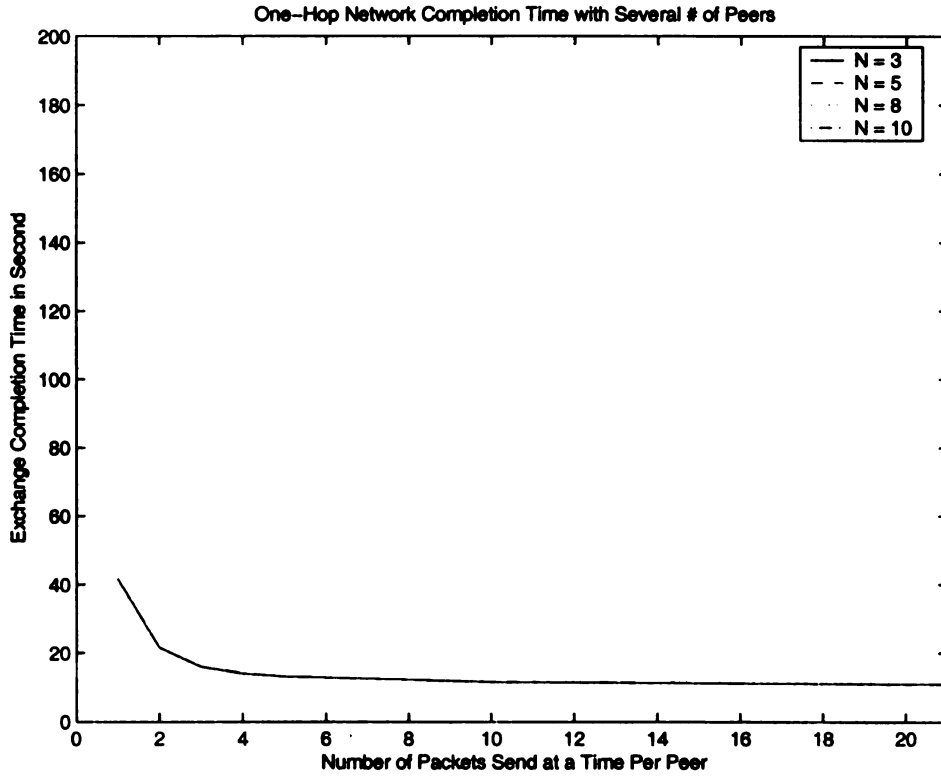


Figure 5.11: Exchange completion time with different number of packets sent at a time per peer

the completion time. As more data packets transmit at a time per each peer, it will take a smaller completion time and reduces the overall DONE packet transmission frequency. Fig. 5.11 and Fig. 5.12 show the simulation result of the completion time by sending different numbers of data packets at a time. The target file size is 1 Mbyte and peers broadcast 512 byte of data packets. Fig. 5.11 illustrates the completion time of a $CHUM_p$ network in which all peers are within the transmission range of others. As the number of data packet sent at a time increases, the completion time to construct the target file decreases. However, as the number reaches 10, the completion time decreases slowly. One interesting point in the figure is that, regardless

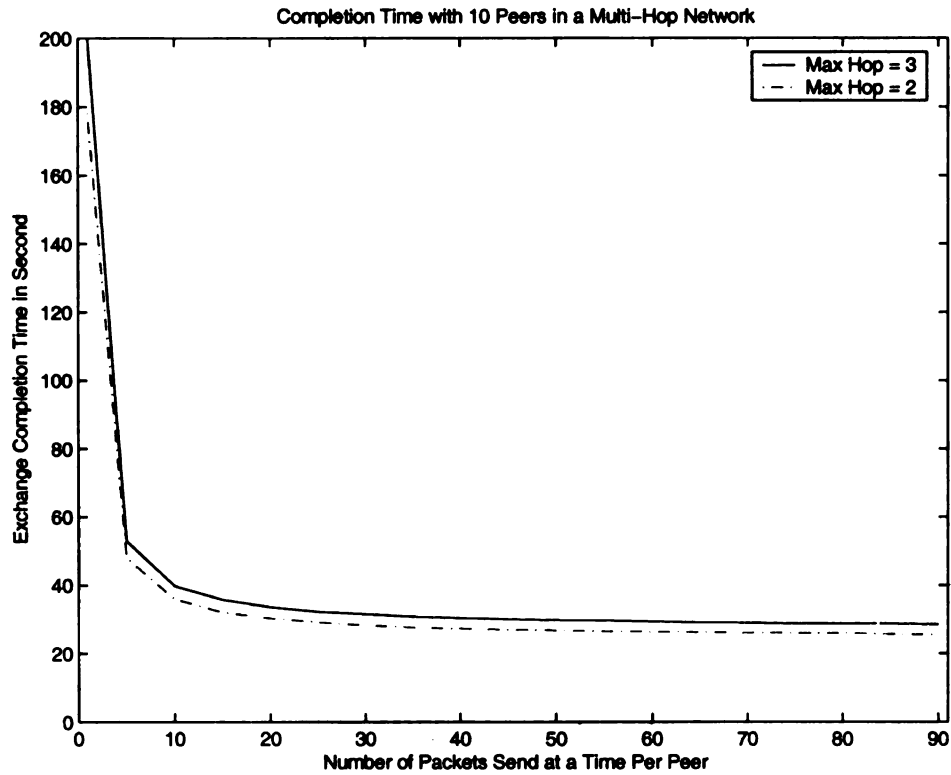


Figure 5.12: Exchange completion time with different network size. Larger transmit-at-a-time value decreases the completion time

of the number of participating peers in the network, it shows a similar completion time for all cases in one-hop networks. It is because both the target file size and the total number of data packet transmission are fixed, and the transmission actions are distributed to the peers regardless of the number of peers. However, as more peers participate, less data is downloaded from the Internet by each peer, which reduces the telecommunication cost. Fig. 5.12 displays the completion time of 10 peers in multihop networks. As the number of hops in a network increases, the completion time also increases. If a peer is allowed to send only one packet at a time, per-peer

distribution produces poor performance. Good performance occurs when each peer broadcasts multiple data packets at a time.

This distribution method does not allow rebroadcasting of data packets. However, the DONE packet is rebroadcasted immediately, if specified in the packet. The sender of the DONE packet needs to check whether the packet is delivered correctly to its one-hop neighbors. After sending the packet, the sender waits for some time expecting a reply from one of its direct neighbors, which may be a data packet or a rebroadcasted DONE packet. If there was no reply within a specified time, for example 100 msec, the peer repeats to transmit the DONE packet. There should be no reply if the network topology changes due to peer movement. This failure can be recovered by receiving new rebroadcasting set information from its associated server.

A per-peer based distribution method also has a data recovery procedure with a bitmap similar to that used in per-packet based method. One difference of using a bitmap between the two methods is that each bitmap is not uploaded to its associated server, but it is broadcasted to each peer's neighbors, which reduces the 3G communication cost. When a peer receives a bitmap from one of its neighbors, it examines the bitmap whether there is any bit reset. If the peer does not have the corresponding data of the bit position, it skips the bit. However, if the peer has the data and the corresponding bit is reset, it modifies the benefit value of the corresponding data to one so that it will transmit the data later when it is the peer's turn.

When a peer receives all data and it sets all bits in its bitmap, the peer transmits a BITMAP packet during its turn, which contains its bitmap with two

other values: bitmap initiating peer's PID and the boolean true value. The BITMAP packet triggers the transmission of the scheduled next peer, and it is forwarded if the rebroadcasting peer set is specified in the packet. After examining the received BITMAP packet, the peer with a turn has three options. First, if it has any data whose benefit value is positive, it broadcasts data packets and discards the BITMAP packet. Second, if it has no more data to send and it does not have the complete data, it replaces the PID and bitmap with its own, and assigns false to the boolean value that indicates some peers are not yet completed. It broadcasts the updated bitmap expecting some neighbors may fill the gap. Third, if it has no data to send and has complete content, it rebroadcasts the BITMAP packet without changing any value. When the BITMAP packet initiator receives the packet from its neighbor with its PID and boolean value unchanged, the peer confirms that all peers have the complete content.

5.4.1 Simulation results comparison

The *ns2* network simulator [130] is used in this comparison. The simulation model assumes each peer downloads portions using its 3G connection that the master server schedules to download, while it exchanges the portions with other peers using its *ad hoc* network. The peers do not experience buffer underrun when they transmit their portions to other peers. The unit packet size of the 3G connection is set to 512 bytes. The *ad hoc* network uses the same size of data packets. The peers download the target file size of 1 Mbytes that consists of 2048 unit packets. Each peer has the

802.11 MAC with the transmission range of 250 meters. The number of peers varies from 2 to 10. Because of the short amount of completion time described in [126], the mobility of peers is not considered. All peers are located in a 400 meter by 400 meter grid unless specified otherwise. In case of per-peer based distribution, each peer transmits the maximum of 90 packets at a time because the packet buffer size is set to 100 in the simulation. Each peer uploads its new neighborhood information in a second. In all cases, the *ad hoc* network is connected.

Fig. 5.13 and Fig. 5.14 illustrate the basic characteristics of the two distribution methods depending on the number of hops and the number of peers in the network. In Fig. 5.13, the simulation uses simple topologies. N peers construct a network of $N-1$ hops. In order to measure the completion time of a 4-hop network, 5 peers are located in a line with the distance of 200 meters between any two neighbor peers. As illustrated in the figure, per-peer based distribution shows scalable performance as the network size increases. Per-packet based distribution, however, takes more time when the maximum number of hops reaches four or more. It is because, as the packet exchange proceeds, some data packets and triggering DONE packets may collide. The scheduled next peer loses a chance to transmit. In the per-packet based method, packets may be lost due to collision or peer movement. If some rebroadcasting peers move to other places, the network might stop exchanging packets. The transmitting peer cannot distinguish the reason for the packet lost between packet collision and peer movement. The packet lost can be recovered after receiving new topology information from its associated server that takes one second in this simulation. The per-peer based

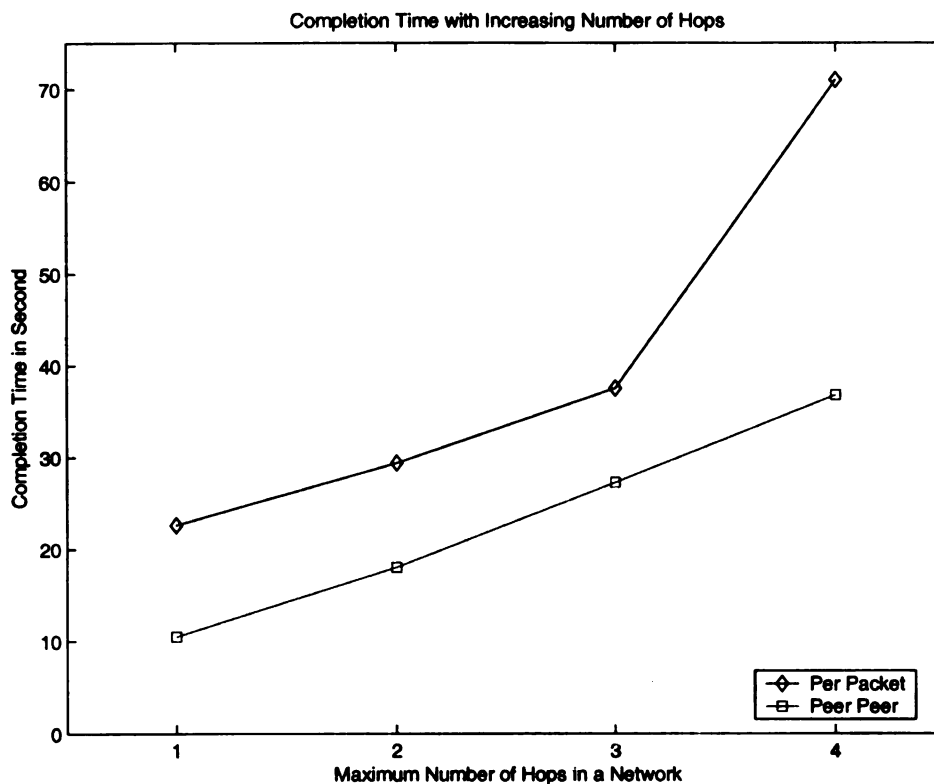


Figure 5.13: Exchange completion time by increasing the number of hops

method hardly experiences any packet collision. Overall, the per-peer based method outperforms per-packet based method in terms of the completion time. Fig. 5.14 shows the completion time with a varying number of peers in a 3-hop network. When one peer is in a cluster, the *ad hoc* network has four peers located in a line, each separated by 200 meters. In general, when there are N peers in a cluster, the network has $4*N$ peers participating. In the per-peer based distribution, it shows constant completion time regardless of the number of peers in the network. However, the per-packet based method shows a slight increase in the time as the number of peers increases. The figure indicates the per-packet based method is affected by the density of peers.

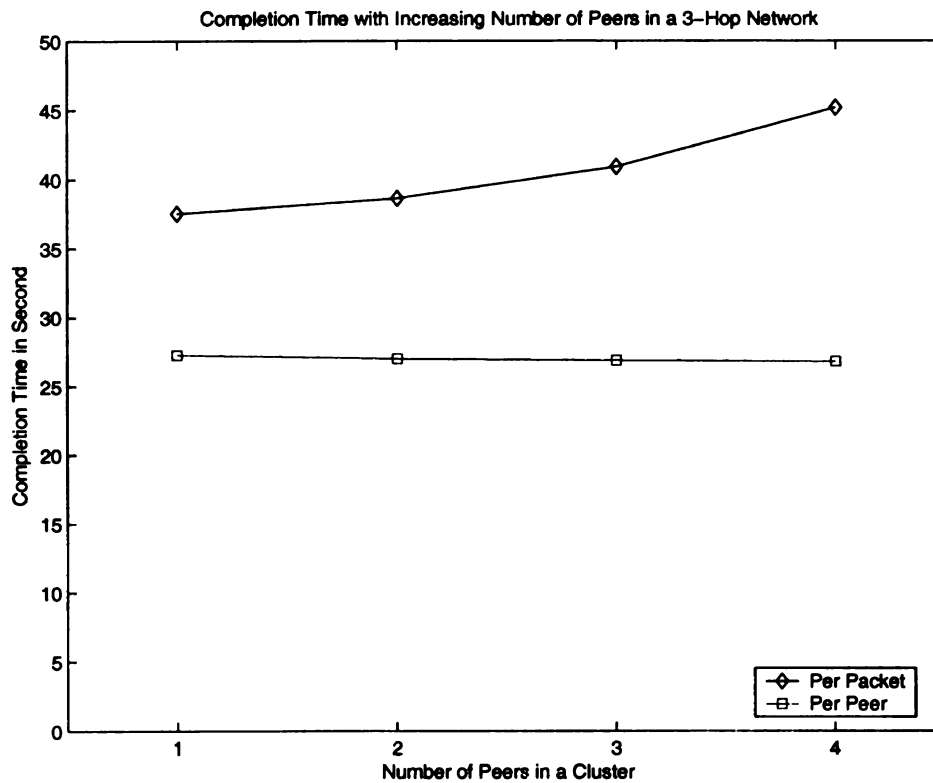


Figure 5.14: Exchange completion time by increasing the number of peers

In large-sized networks, there is a possibility of exchanging data between some groups of peers in parallel. That is, when the transmission ranges of two or more peers do not conflict, they may transmit their content independently with others in parallel. The parallel data exchange may produce faster completion time than that of the per-peer method provided that some requirements are permitted. First, the maximum hop of the network should be at least three. Second, a complex transmission scheduling algorithm is required by the master server because the transmission sequence of peers is irregular. The master server computes the next transmitting peer(s) by collecting the up-to-date data reception status from all peers that upload their bitmaps using their 3G telecommunication links. All peers upload their status

information frequently in order to select the next suitable sender(s) because some data packets might have been lost due to collisions. Both distribution methods use an asynchronous mechanism in which the next transmitting peer knows its turn by the triggering packet, such as the DONE packet. However, in the parallel exchange method, the next transmitting peer may resume its transmission by the signalling packet either from the master server or from its predecessor(s). Receiving the signal from the server incurs some overhead because peers become senders several times during the distribution process. If they complete downloading their assigned portions first and then exchange them later, the overhead will be minimized. However, content distribution over the *ad hoc* connection will be postponed until a peer downloads its portion of the content. If the next transmitting peer receives the resume signal from its predecessor(s), there is a possibility of losing the signalling packet because peers may broadcast packets at the same time and cause collisions. Parallel data exchange may outperform the per-peer method in terms of the completion time in large-sized networks at a higher cost of using the 3G telecommunication link. The per-packet and per-peer methods are suitable for small or medium-sized networks with low telecommunication cost.

Fig. 5.15 displays the average number of packets used by the peers that may have a fee charged by a telecommunication provider. Each value in the figure is the average of 30 runs. As the number of peers increases, the number of fee-based packets decreases substantially. However, an additional peer only reduces a marginal cost when there are already several peers, whereas, the new peer receives the same

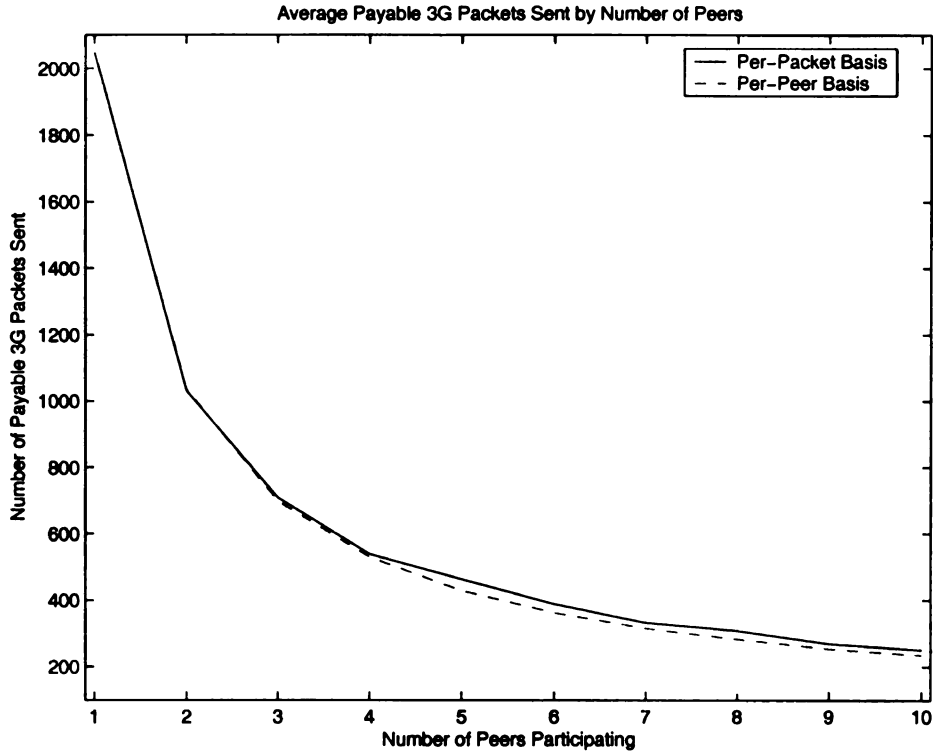


Figure 5.15: The number of packets on the 3G link by varying the number of peers cost reduction. Approximately 90% of the telecommunication cost is saved with as few as 10 peers. The per-packet based method uses slightly more packets on the 3G telecommunications link than that of per-peer based method. This is due mainly to the recovery process in which each peer uploads its bitmap reporting any data gaps to its associated server. Per-peer based distribution resolves the gap by exchanging bitmaps with its neighbors over the cost-free *ad hoc* connection. The completion time of each simulation is shown in Fig. 5.16. Both horizontal lines are the average of 30 runs and the short vertical lines represent the standard deviation of the 30 results. The average completion time increases slowly as more peers participate. This is due to the increasing number of hops in the network. All *ad hoc* networks in this simulation

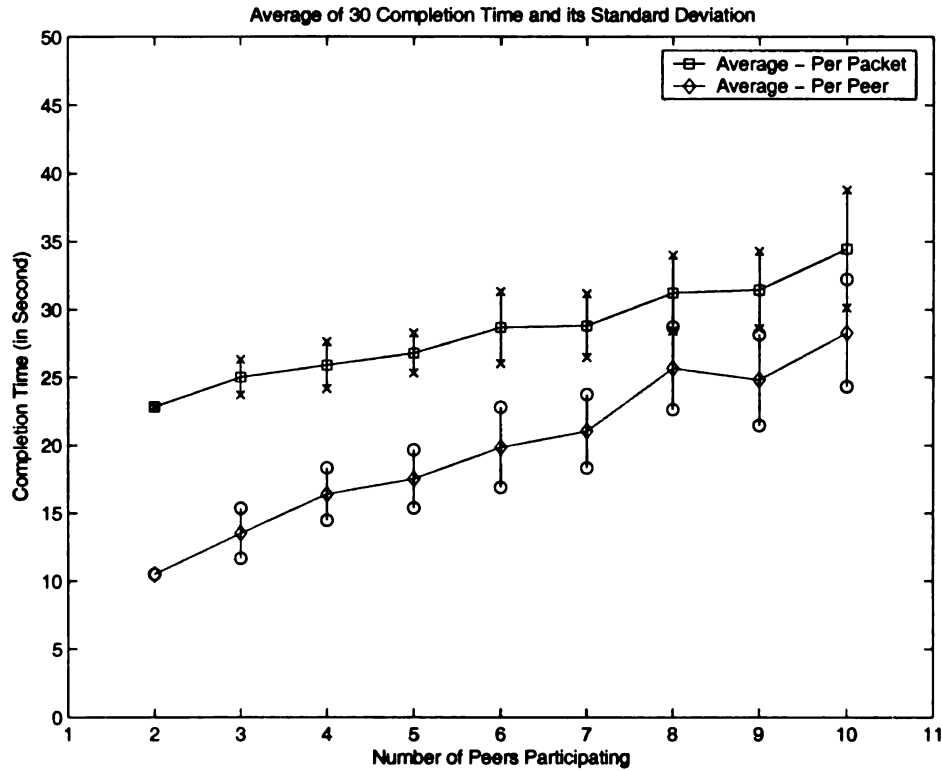


Figure 5.16: Exchange completion time by varying the number of peers

are connected. That is, each peer is connected to at least one other peer. With small number of peers, they locate a part of the simulation grid of 400 square meters. As the number of peers increases, they may disperse over the grid, which may increase the number of hops. More hops cause more time to complete the distribution. Per-packet based distribution may also suffer from packet collision as the network becomes crowded. According to the graph, per-peer based distribution completes the content distribution faster than that of per-packet based one. One important point is that as the number of peers increases, the difference of the two completion times decreases. This result indicates that the number of hops in a network is a more dominant factor than the density because per-peer based distribution hardly experiences any collision.

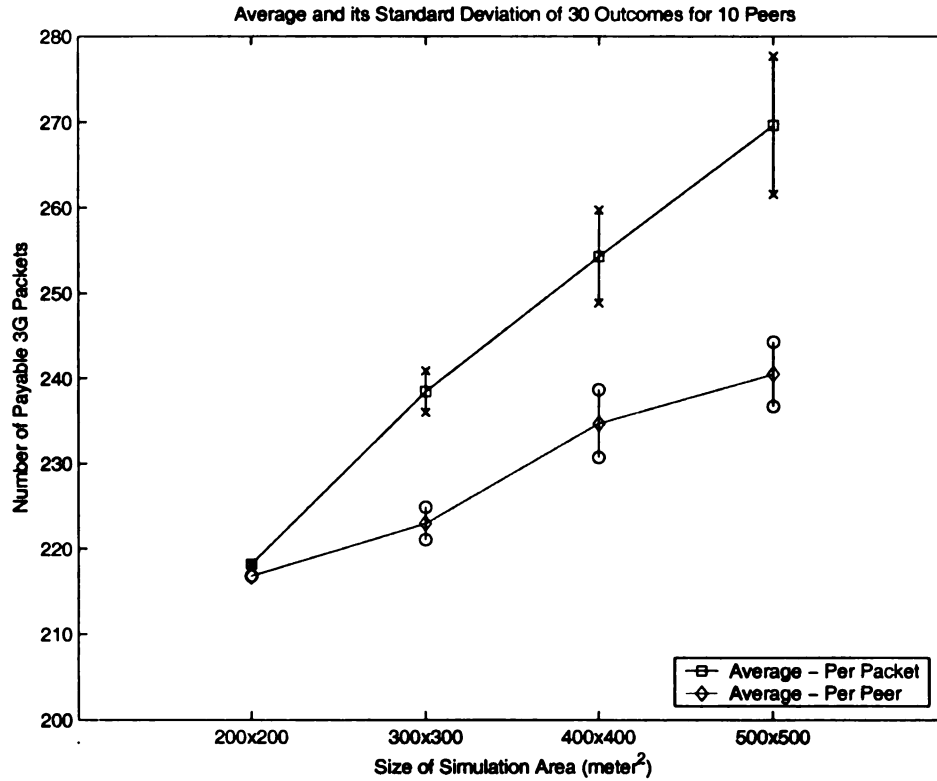


Figure 5.17: The number of packets on the 3G link by varying the size of area

Among the 270 network topologies in a 400 square meter grid, the maximum number of hops was 3. Suppose larger simulation grids are used and the largest hop reaches 4 or more. The completion time differences become larger as given in Fig. 5.13 and Fig. 5.14.

As mentioned before, the $CHUM_p$ network is greatly influenced by the number of hops in a network. A larger simulation area leads to larger number of hops. When all peers are located within the transmission range of each other, no collisions are expected because only one peer transmits at a time. Fig. 5.17 and Fig. 5.18 illustrate how the performance changes due to the change of the grid size. The simu-

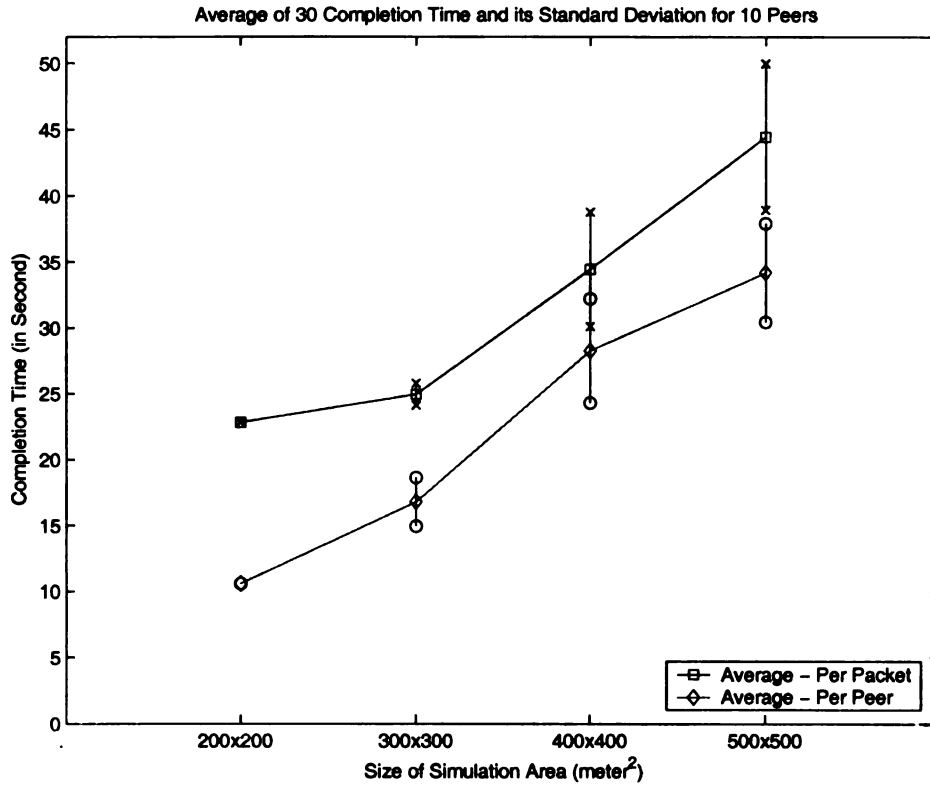


Figure 5.18: Exchange completion time by varying the size of area

lation runs with 10 peers and repeats 30 times with 30 different network topologies. Fig. 5.17 shows the number of packets on the 3G telecommunications link as the size of simulation area increases. A larger area may result in a larger number of hops in a network, and more time to take to complete the content distribution. Because peers upload their neighborhood information periodically, more completion time increases the number of packets on the 3G link as well. In addition, peers upload their bitmaps and download the recovery instructions for per-packet based distribution, which consumes packets on the 3G link. Due to more collisions in a larger area, the per-packet based method uses more 3G packets than that of the per-peer based method. Fig. 5.18 shows the completion time by varying the simulation area size. The largest number of

hops was 3 among 120 different topologies used in the figure. As the simulation size increases, so does the number of hops, which causes longer completion time. Overall, the per-peer based distribution method outperforms the per-packet based method in terms of the completion time, and uses less packets on the 3G link.

5.5 Summary

This chapter introduces the *CHUM_p* network in which mobile peers save telecommunication cost by sharing their partially downloaded data with other peers. All peers consent to download a specified target file located in the Internet using their fee-based WWAN connection. The peers are associated with their servers, and one of the special servers, called the master server, performs peer membership management, download scheduling, and data distribution scheduling. Each participating peer downloads only a portion of the file over the cost-based connection, and distributes its downloaded portion to all other member peers over the cost-free *ad hoc* network so that all participating peers can construct the complete target file. This chapter presents two content distribution methods: per-packet and per-peer based distribution methods. Both distribution methods do not use any specific routing or multicasting protocols, but utilizes only the inherent wireless broadcast mechanism. The simulation results show that per-peer based method outperforms per-packet based method. In addition, approximately 90% of the telecommunication cost is saved with as few as 10 peers in both distribution methods.

Chapter 6

Anonymous Sharing in CHUM networks

6.1 Introduction

The previous chapters describe CHUM networks in which the participating peers cooperate to download streaming or non-streaming data from the Internet at the same time. That is, the requested content is stored in the Internet, and one or all peers download the partial or complete content from the Internet in order to share it with other peers. This chapter introduces a CHUM network, labeled $CHUM_a$, in which some peers may acquire partial or complete content from the existing $CHUM_a$ network, and only the missing fraction of the content, if exists, can be downloaded from the Internet. A peer may join a $CHUM_a$ network at any time, and it may request its wanted content from the network. The peers already involved in the $CHUM_a$

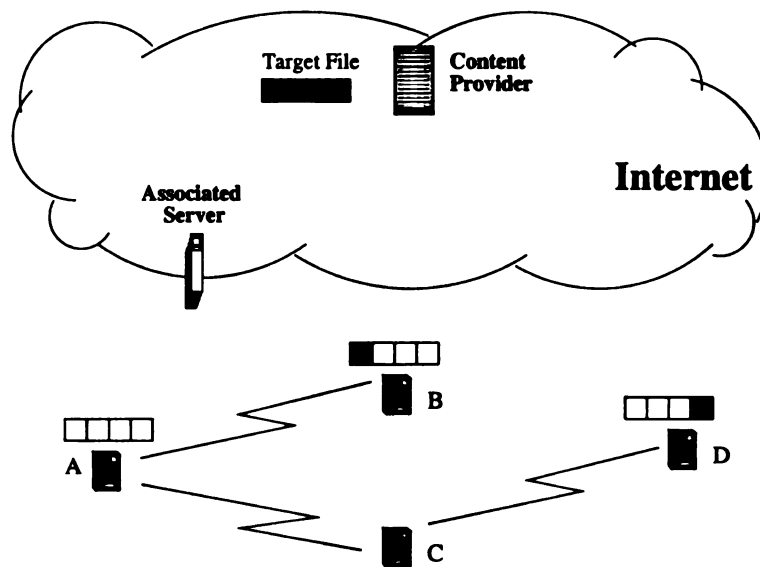


Figure 6.1: Two peers transmit their partial content to requesting peer

network cooperate to share their stored content with the requesting peer. Suppose several groups of students are studying outdoors and each group may download its assigned topic from the Internet. One of the mobile devices in a group, say a group leader or a teacher, may download the content first. All students in the group may copy the content from the leader device with no cost paid. When a group of students changes its topic, they can search the next topic from their neighborhood first and, if not found, then retrieve it from the Internet. Similar example can be found in outdoor galleries. In order for a mobile user to find the description and background of a displayed work, the user may search the related information from its neighbors first and then from the Internet next, expecting that some nearby users may already have downloaded the content. In a sports stadium, several spectators may download information about the players, the game records, the replays, etc. If one of them already have downloaded the content, nearby mobile users can share the content

using a cost-free *ad hoc* connection, rather than paying a cost for the 3G connection. It is also possible that a user may browse a list of files stored in the nearby *CHUM_a* network, and selects one of the files listed. The stored file will be delivered from the *ad hoc* connection to the user.

Fig. 6.1 displays a situation in which a mobile device A wants to acquire the target file, originally stored in a content provider (CP) on the Internet. Suppose devices B and D have some segment of the target file. The reason for the devices to store only some part of the file may be that they initially stored the whole content and reallocated some memory segment taken by the file. Multiple senders, both B and D, transmit their segments of the target file to the receiver A. The requesting device A does not need to know who is the actual source of the segment. In addition, if devices A and D are out of transmission range of each other, some intermediate device, such as C, will forward the content for device A. After collecting the partial segments, device A requests the remaining segment to the proxy server, which is illustrated in Fig. 6.2. The proxy server fetches the whole content of the target file, but delivers only the requested segment(s) to device A in order to reduce the 3G communication cost.

There are several challenges to solve for content fetching from other devices. First, the network should provide a mechanism by which a peer can discover its needed content, if it is stored by other peers in the network. Peers should identify the content with a unique name in the network. URL-based naming provides a way of specifying a file uniquely on the Internet [116]. However, it is not efficient for

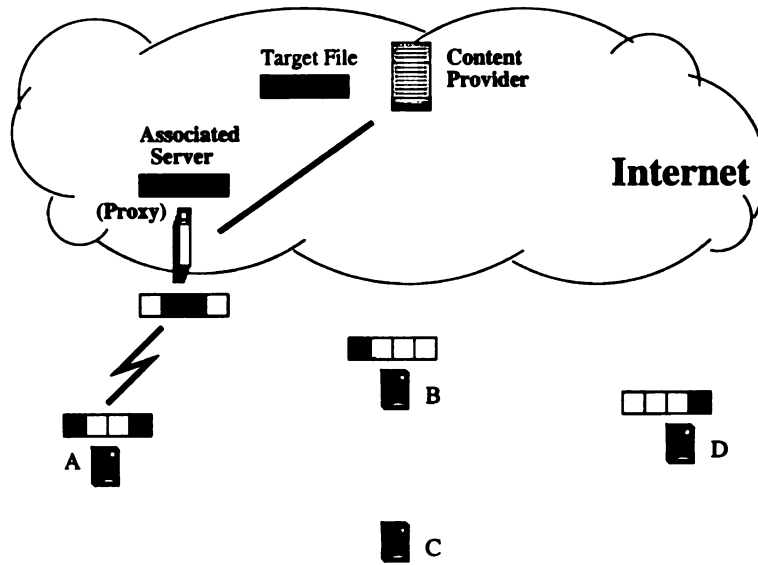


Figure 6.2: The requesting peer downloads remaining content from $CHUM_a$ proxy peers to keep the long URL strings for managing files. It is rather convenient that peers use integers describing files on the Internet, and $CHUM_a$ servers map the URL-based files to integers. The second challenge comes from the distribution from the peer(s) storing the content to the requesting peer. If the requesting peer is out of transmission range of the source peer(s), some intermediate peer(s) between them should forward the content. When multiple source peers may transmit their content, the distribution mechanism should guarantee to avoid collision of packets from the peers. In addition, if one peer contains the whole content of the requested file and another peer stores only the half of it, the transmission of the later peer needs to be declined. Furthermore, the requesting peer does not need to know who is the source peer, and the source peer also has no knowledge about the requesting peer. The anonymous connection between the two peers makes other peers difficult to guess who is talking to whom.

This chapter describes a $CHUM_a$ network in which each mobile peer associates with a $CHUM_a$ server in the Internet. Each associated server collects neighborhood information and the sharable file status of its associated peer. One of the associated servers plays a role of the master server, and it gathers all partial neighborhood information and creates a logical network that mirrors the global view of the *ad hoc* network. In addition, the master server maintains file status information that represents files stored by the participating peers.

6.2 $CHUM_a$ network operation

6.2.1 File naming and mapping

URL-based file naming provides a way of uniquely identifying a file on the Internet [116]. Peers use the URL-based file name to request an existence of a file and to fetch a file in the $CHUM_a$ network. It is, however, inefficient for peers to use a long string of the URL-based file name. In the $CHUM_a$ network, the master server performs a mapping of a peer's requesting file to a unique number within the $CHUM_a$ network. The master server maintains the mapping information, and answers any queries from peers. Peers use FIDs when they request a file existence in a $CHUM_a$ network or when they transmit their stored file segment(s) to other requesting peers. That is, initially, all file names are specified by URL-based naming, but peers and servers use integer FIDs for specifying files internally in a $CHUM_a$ network.

A peer has two ways of requesting a FID in a $CHUM_a$ network. First, the peer may request a FID by sending an URL-based file name to its associated server that passes the name to the master server. The master server retrieves the file name from its FID table. If the file exists in the table, the server returns the corresponding FID with the percentage information how much portion of the file is stored in the network. In the opposite case that the file information is not in the table, the master server inserts a new entry in the table. It, then, creates a new FID and writes both the FID and the file name on the entry. It also puts zero to the percentage column because the file has never been downloaded before in the network. The FID and the percentage value are returned to the requesting peer. If the percentage value is larger than zero, the associated server of the requesting peer asks the master server which peer contains the file. However, if the received percentage value is zero, it indicates that the $CHUM_a$ network does not have the file, and the peer will download the file itself from the Internet. After downloading the file, the peer uploads the file status to the master server informing that it stores the whole content of the file. The second way of acquiring a FID is that the peer may request a list of file names stored in the $CHUM_a$ network. The list contains a FID, a corresponding URL-based file name, and a percentage of the file stored. The peer browses the list and selects a file that is interested. As the peer acquires a FID of an interesting file, it requests the file with the FID to the master server via its associated server.

6.2.2 File distribution mechanism

When the master server receives a request with a FID from a peer, it searches the matched FID from the FID table. No match triggers a notification to the peer via its associated server that the $CHUM_a$ network does not contain the requested file. Suppose, the master server finds one or multiple entries of the file in the table. The server checks the timestamp of each entry and decides whether to fetch up-to-date information about the current status of the file stored by specified peers. That is, the $CHUM_a$ network operates mainly on-demand basis. Peers report the file storage status of themselves to the master server periodically but infrequently, such as once in several minutes. After examining the timestamp of the FID table entry, the server picks peers whose timestamps are expired in order to refresh their status information. Selected peers reply their up-to-date file storage status as well as their neighborhood information. The master server decides which peer becomes the content sender based on the received information. If one peer contains a complete content of the file, other peers will be declined to be senders. If several peers have the whole content, the peer of the smallest number of hops from the requesting peer will be selected. The master server selects all peers that have a unique portion of the file.

In order for the master server to select a peer storing a portion of a file that other peers do not have, the server keeps file status information that each peer contains which segment of a file. The segment can be represented as a combination of the base address and the offset. Fig. 6.3 displays a snapshot of the FID table of Fig. 6.2. The first column is not a data but an index of the two dimensional array. The value of the

| Index | FID | PID | Timestamp | Base | Offset | Next | |
|-------|------|------|-----------|------|--------|------|------|
| | | | | | | | |
| [6] | 3 | A | 1235 | 0 | 25 | 7 | |
| [7] | 3 | A | 1235 | 75 | 25 | -1 | |
| [8] | 3 | B | 917 | 0 | 25 | -1 | |
| [9] | 3 | C | 875 | 75 | 25 | -1 | |
| | | | | | | | |

Figure 6.3: Snapshot of FID table representing Fig. 6.2

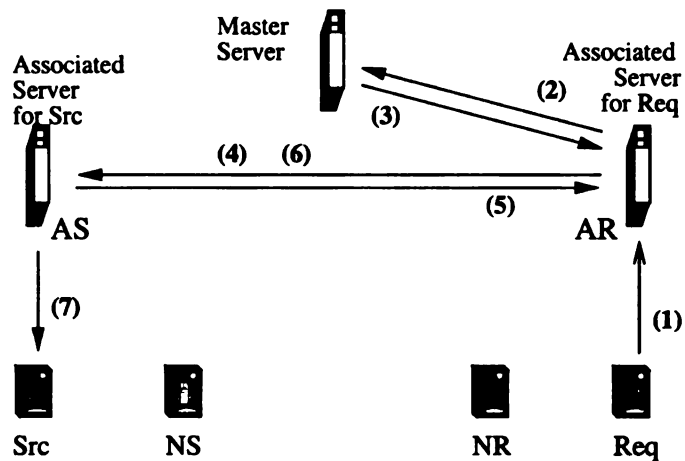
last column labeled *Next* points to the next segment entry index of a FID in the table. The negative value implies the end of the segment of the specified file. The smallest base value of a FID indicates the first segment stored by a peer of the PID. The figure assumes that the file size is 100. By examining both the base and the offset of a FID, the master server is able to select peers that contain a unique segment of the file. If the timestamp exceeds a threshold compared with the current time value, the server requests peers that have the file segments to upload their current file storage status and the neighborhood information. The entry will be updated when new segment information arrives.

The $CHUM_a$ network provides privacy between the source peer and the requesting peer. That is, the requesting peer does not know who is the actual source peer. Moreover, it is difficult for any neighboring peers in the $CHUM_a$ network to perform traffic analysis. The purpose of traffic analysis is to help deduce which peer is talking to whom by analyzing traffic patterns instead of the data that is transmitted. Even though the data is encrypted, traffic analysis may detect the two peers com-

municating. The provisioning of privacy requires an anonymous connection between the source and the requesting peer. The master server becomes a trusted entity to distribute the secret key set to peers for data encryption/decryption, and helps to build an anonymous connection. The associated server of the requesting peer performs the detailed anonymous connection setup. Only the associated server of the requesting peer knows about the actual source and the requesting peer. No other associated servers nor the master server know both peers. Furthermore, no peers can guess correctly who is talking to whom including the source and the requesting peer themselves. Two intermediate peers are involved in the anonymous connection between the source and the requesting peer. The onion routing [131] indicates that a single intermediate node is sufficient to complicate traffic analysis.

Anonymous connection setup

The $CHUM_a$ network provides an anonymous connection between the source peer and the requesting peer. The anonymous connection requires a participation of some intermediate peers which makes traffic analysis difficult. Fig. 6.4 shows the overall sequence of the anonymous connection setup. The associated server of the requesting peer, called the AR server, selects one peer from the neighbor list of the requesting peer, called the NR peer. The AR server, then, requests the associated server of the source peer, called the AS server, to pick one peer from the neighbor list of the source peer, called the NS peer. The communication between the AR server and the AS server may use an existing Internet-based anonymous connection service such as



- (1) Peer Req wants a file (segment) F
- (2) Associated server of Req, AR, requests F to master server
- (3) Master server confirms that peer Src has the file F
- (4) AR requests a neighbor of Src to AS, associated server of Src
- (5) AS selects NS peer and sends PID of NS to AR
- (6) AR selects NR peer, creates CIDS, and sends CIDS to AS
- (7) AS passes CIDS to construct an anonymous connection

Figure 6.4: Overall sequence of anonymous connection setup

the onion routing. The anonymous data path starts from the source peer to the NS peer to the NR peer, and finally to the requesting peer. Though all peers are located within the transmission range of others, three transmissions will be needed to send a data packet from the source to the requesting peer. Furthermore, if the source peer is located on the route between the NS and the NR peer, the source peer may become one of the forwarding peers. The route from the source peer to the requesting peer is known except the path between the NS and the NR peer. The route between the two intermediate peers can be found using a well-known *ad hoc* routing protocol, such as AODV [29].

| Dest. Address | Crypt for Padding | Crypt for Forward Data | Crypt for Backward Data |
|------------------|----------------------|---------------------------|----------------------------|
| NS | Sym Key P1 | Sym Key E1 | Null |
| NR | Sym Key P2 | Sym Key E2 | Sym Key E1 |
| Req | Sym Key P3 | Sym Key E3 | Sym Key E2 |
| Null | Sym Key P4 | Null | Sym Key E3 |
| Padding | | | |

Figure 6.5: CIDS example of Fig 6.4

The AR server knows all about the source and the requesting peer, and it has the responsibility to construct the anonymous connection. No other servers including the master server nor any peers do not know completely about the source and the requesting peer. The AR server creates a connection information data structure (CIDS) that is similar to a collection of onions in the onion routing [131]. The CIDS is a four-layered data structure. Each layer has the same format and the same size. One layer contains a destination peer address and three keys for its bi-directional virtual connection depicted in Fig. 6.5. Each layer, except the innermost or the fourth layer from the top, represents one of the three virtual connections made by the four peers: the source, the NS, the NR, and the requesting peer. From the outermost to the innermost, each layer is used by each of the four peers in that order.

The CIDS is delivered from the AR server to the source peer via the AS server. When one of the four peers receives the CIDS, it applies its private key on the fixed-sized top layer to read the virtual connection information such as a destination address and keys, because the layer is encrypted using the public key of the peer. As the peer

knows information of the next peer and the keys, it has to deliver the CIDS to its next peer. The anonymous connection information for the next peer is hidden inside the padding of the CIDS. Before sending the CIDS, the peer removes the top layer of the CIDS and applies the first key on the remaining layers (padding) of the CIDS. The new CIDS will be transmitted to the next destination peer in order to build a bi-directional virtual connection. For example, when the source peer receives a CIDS, it applies its private key on the fixed-sized top layer to acquire the address of the next virtual hop peer (the NS peer) and three keys. The second key will be used to encrypt data transmitted to the NS peer. The third key is used for backward transmission, but it is useless for the source peer. The source peer removes the top layer and applies the first key on the remaining part of the CIDS. The source peer sends the decrypted CIDS to the NS peer so that they construct a virtual socket connection between the source and the NS peer. The NS peer performs a similar task and creates a virtual connection with the NR peer. The backward key of the NS peer is the same as the forward key of the source peer. When the requesting peer receives the CIDS from the NR peer, it finds that the peer itself is the final destination, because the destination address is set to a null value. As a result, a series of actions generates a bidirectional anonymous virtual connection between the source and the requesting peer with the aid of both the NS and the NR peer.

The AR server, initially, should create the innermost layer of the CIDS first. Then, it decides a size of padding right after the innermost layer. The innermost layer will be encrypted using the public key of the requesting peer, and the padding will

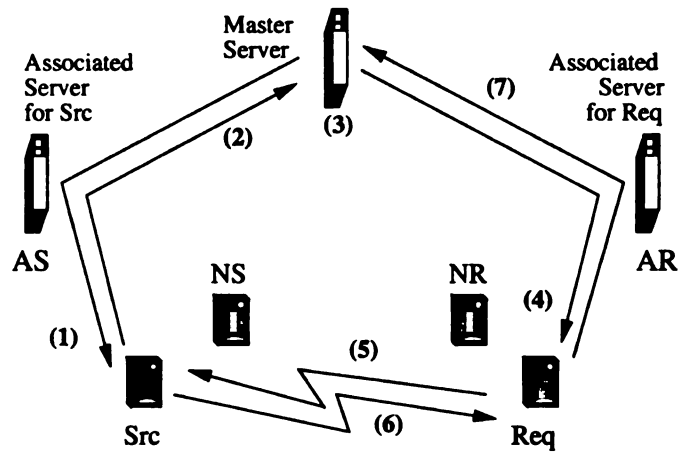
be encrypted with the key P4 in Fig. 6.5. The AR server, then, creates the second innermost layer for the NR peer. This layer will be encrypted using the public key of the NR peer, and the remaining part (the layer for the requesting peer plus the padding) will be encrypted with the key P3. This process repeats until the outermost layer for the source peer is encrypted by the public key of the source peer and the remaining part (the three inner layers plus the padding) is done with the symmetric key P1.

File distribution

As the anonymous connection is established, the source peer receives an acknowledgement packet from the requesting peer over the backward connection. The source peer, then, starts transmitting the file (segment) to its next virtual hop peer, the NS peer. The route discovery may not be required between the source and the NS peer, because they are within one hop. If one of the two peers move away and the connection is broken, the source peer should recover the path by using the path recovery procedure of the used *ad hoc* routing protocol. When one of the two peers could not be discovered, the error is reported back to the AR server. The server repeats the construction of the new CIDS with a newly selected intermediate peer. The route between the NS and the NR peer needs to be discovered prior to data transmission. When the NS peer receives a CIDS, it should know the next virtual hop peer, the NR peer, and executes a route discovery procedure of the *ad hoc* routing protocol. The NS peer relays the received data from the source peer to the NR peer over the

found route. The NS peer becomes the packet source of the relayed data in view of the NR peer. The NS peer reconstructs a network-level header without changing the payload that is received from the source peer, and transmits the packet to the NR peer. Similarly, the NR peer unicasts the received data to its destination peer, the requesting peer. The backward route from the requesting peer to the source peer is used when the requesting peer needs to send acknowledgements back to the source peer. If the *ad hoc* routing protocol does not provide a backward route while constructing a forward route, the NR peer needs to discover an alternate route to the NS peer.

As the requesting peer completes receiving the file (segment), it should verify the integrity of the file content and needs to register the containment of the file to the master server. The overall procedure of the content verification and the registration is shown in Fig. 6.6. Before data transmission, the master server checks the integrity of the source peer's stored content. The source peer executes a message digest algorithm, such as MD5 [129], to generate a hash value using the initial digest given by the master server. The master server also uses the same initial digest to compute the final hash of the file segment. If both the received hash value and the computed value are matched, the master server passes the initial digest and the final hash value to the requesting peer via the AR server. The requesting peer is able to verify the received content using the two values. Sometimes, the requesting peer may receive several segments of a file from several peers. The peer may need to combine the segments into one or more segments. Furthermore, the peer may download the missing portion



- (1) Master server verifies the existence of file (segment)
- (2) Source peer replies with a hash value of the requested file
- (3) Master server checks correctness of the hash value
- (4) If correct, master server passes the value to requesting peer
- (5) Requesting peer acknowledges to build anonymous connection
- (6) Source peer transmits the file over the anonymous connection
- (7) Requesting peer computes and uploads a hash value of the file

Figure 6.6: Overall procedure of content verification and registration

from the Internet, if needed, in order to construct a whole file. The peer generates an initial digest and computes a final hash value for each segment or the whole file. The peer, then, uploads the file status information including the base address and the offset of each segment (the file), and the initial digest and the final hash value of each segment (the file). The master server verifies the information, and registers the file information on the FID table.

6.3 Simulation results

This section describes the performance overhead of the anonymous connection using the *ns2* network simulator [130]. The anonymous connection incurs an overhead, be-

cause it needs larger number of hops between the source peer and the requesting peer than that of the direct communication that has no anonymous feature. The simulation model in this chapter assumes that the two intermediate peers are selected and the anonymous connection is established between the source peer and the requesting peer in advance. The source peer has a file segment of 500 Kbyte, and it transmits the file using TCP. Each peer has the 802.11 MAC with the transmission range of 250 meters and the transmission speed of 2 Mbps. The number of participating peers varies from 4 to 20. All peers are located in a 600 square meter grid unless specified otherwise. Because of the short amount of the completion time, the mobility of peers is not considered. In all cases, the *ad hoc* network is connected. One of the three TCP packet sizes is used (256, 512, and 1024 byte), and the simulation uses 1024 byte of packets unless specified otherwise. The simulation adopts three methods of data transmission. The first transmission method, labeled “Direct” in the following figures, does not use any anonymous connection. That is, the packets are transmitted from the source peer to the requesting peer over the shortest path that is discovered by the *ad hoc* routing algorithm (AODV). The second method, labeled “1-Hop”, selects two intermediate peers for the anonymous connection, one from the one hop neighbors of the source peer, and the other from the one hop neighbors of the requesting peer. The third method, labeled “Random”, chooses two intermediate peers for the anonymous connection randomly excluding the source peer and the requesting peer. The overhead is measured by the completion time of transmitting 500 Kbyte file at which the source peer receives the last TCP ACK packet from the requesting peer.

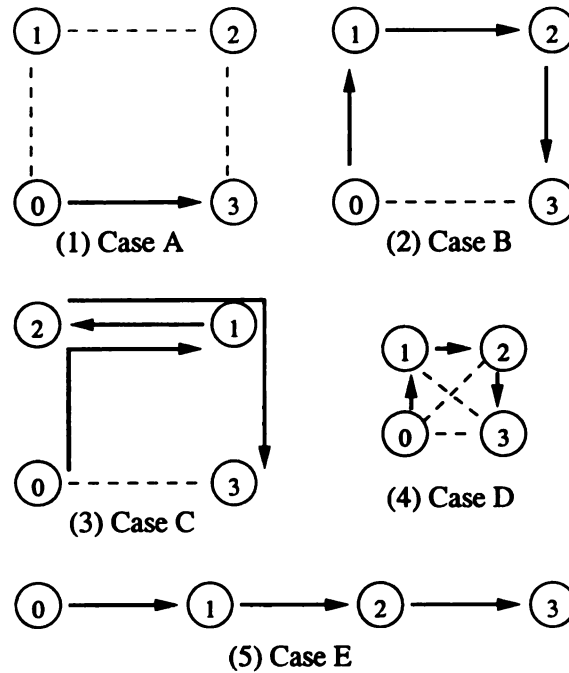


Figure 6.7: Five simple topologies with several routes from peer 0 to peer 3

Fig. 6.7 shows five simple topologies consisting of four peers labeled from 0 to 3. The topologies assume that peer 0 is the source and peer 3 is the destination. Moreover, peer 1 is the next anonymous connection peer from peer 0, and peer 2 is the next from peer 1. The case A use no anonymous connection and the source peer communicates with the requesting peer directly. The case B uses an anonymous connection and it needs three hops to transmit packets from the source to the destination. This is one of the typical case of 1-hop anonymous connection. The case C is an example of the random selection, because peer 1 is not a one-hop neighbor of peer 0, so is peer 2 not from peer 3. In the case A, B, and C, each peer is directly connected with only two other peers. However, in the case D, all four peers are located within on hop transmission range. This case also establishes an anonymous connection. All

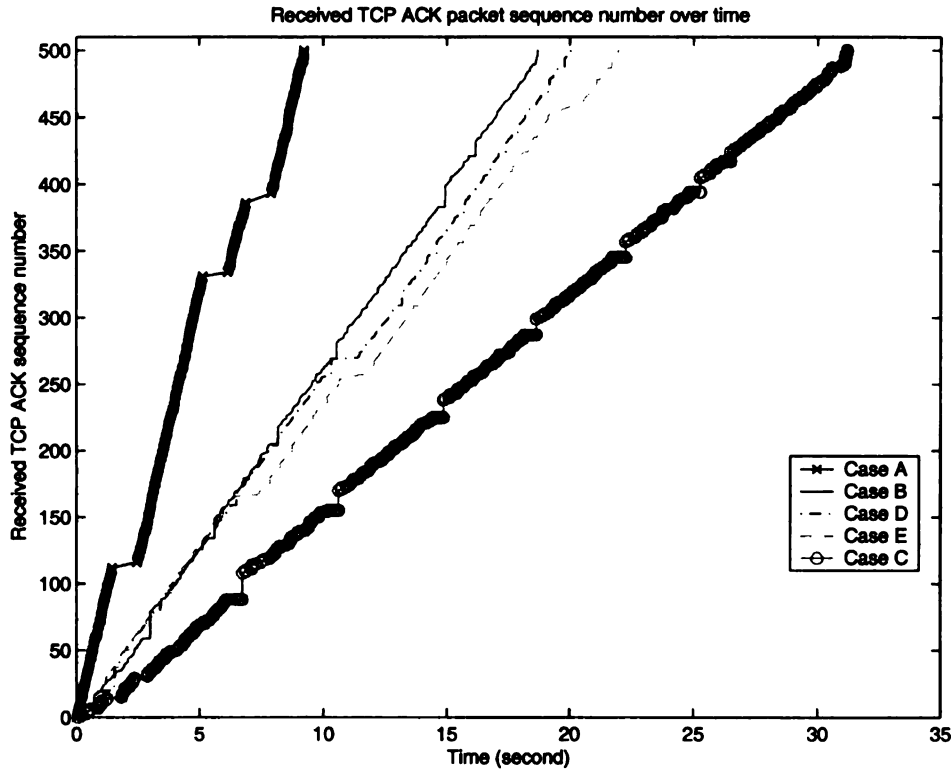


Figure 6.8: Transmission completion time on five different cases

peers are placed in a line in the case E. This case assumes no performance overhead of the anonymous connection, because the packets travel the same path with and without the anonymous feature. Fig. 6.8 displays the sequence of times receiving 500 ACK packets by peer 0 when peer 0 transmits 500 1024-byte TCP packets to peer 3. The case A completes its transmission in less than 10 seconds. The case B, D, and E takes about more or less than 20 seconds. The case C finishes its transmission taking more than 30 seconds. This figure implies that when the source peer and the requesting peer are located in proximity, the anonymous connection overhead becomes relatively larger than without it. In addition, the random selection of intermediate

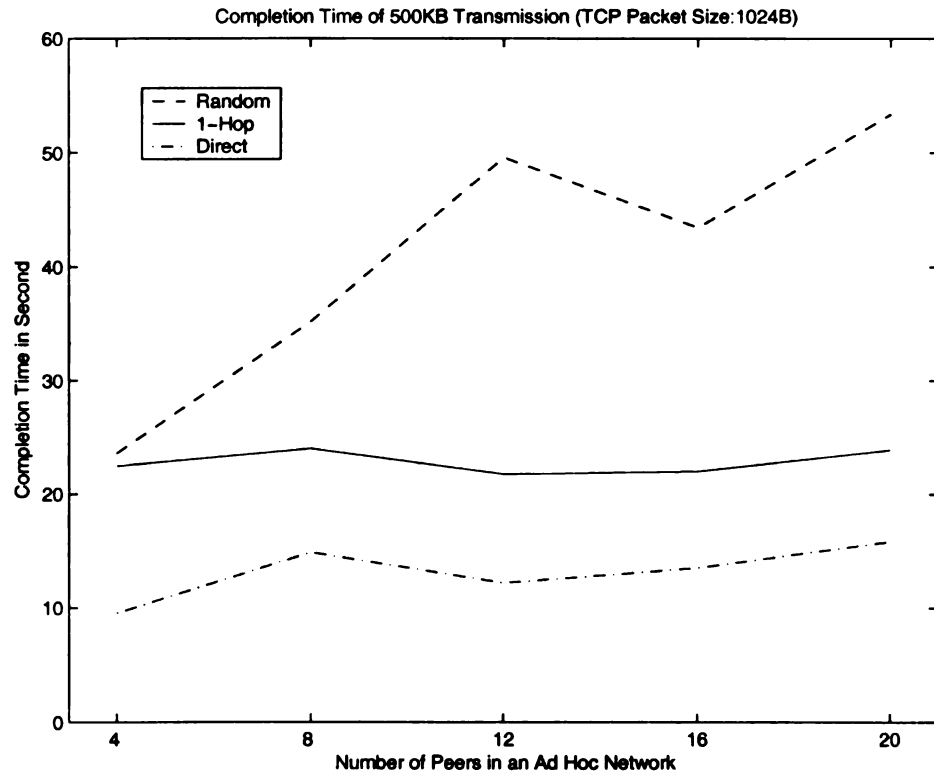


Figure 6.9: Completion time by varying the number of peers

peers may cause relatively substantial overhead compared to the transmission with 1-hop anonymous connection.

Fig. 6.9 illustrates the completion time of receiving the 500th ACK packet when different number of peers are participated in the *ad hoc* network. This figure and the following figures display the average of ten simulation runs from the independently generated random topologies. Both the 1-hop and the direct method show similar completion time patterns, but not for the random method. As the number of peers increases, so does the completion time in the random method. However, in the 12-peer case, some topologies generate relatively larger completion time, which takes

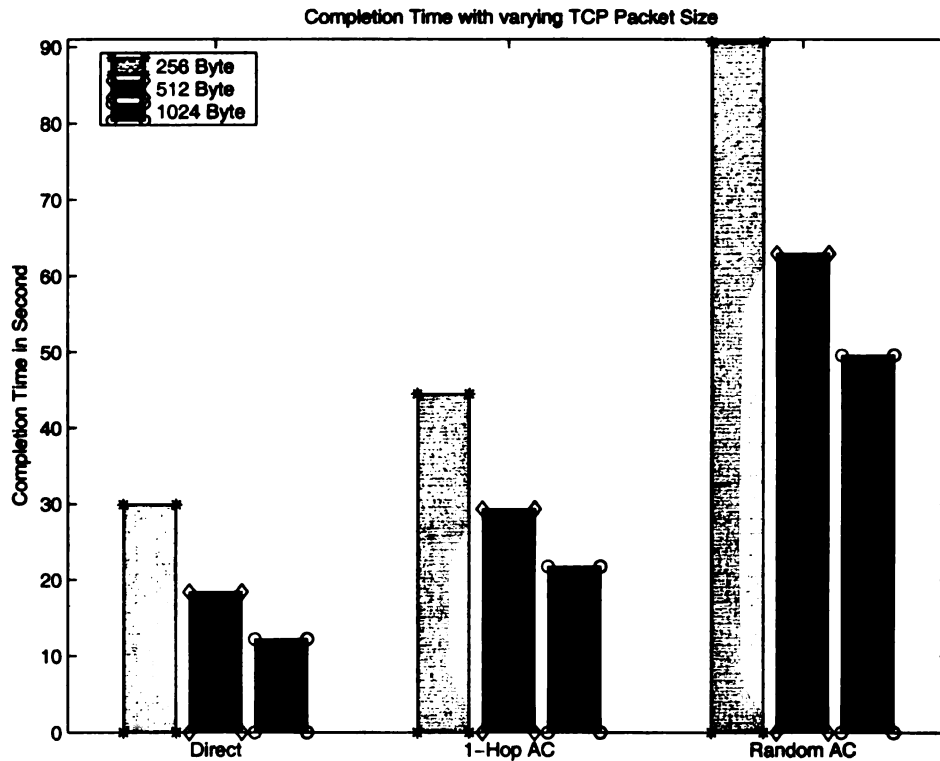


Figure 6.10: Completion time of three different transmission method

more time than that of the 16-peer case. In the 8-peer case, some topologies have longer path from the source to the destination than other number of peer cases, which causes relatively longer completion time for the 1-hop and the direct method than other number of peer cases. In the 20-peer case, the 1-hop method takes 50% more time to complete the transmission than that of the direct method. Fig. 6.10 shows the completion time differences when the size of each TCP packet varies from 256 byte to 1024 byte in the 12-peer case. If the packet size is set to 256 byte, the sender needs to transmit 2000 packets. When the packet size is 256 byte, the 1-hop method needs 50% more time to complete than that of the direct method. In the case of 1024 byte, the average of 78% more time is required.

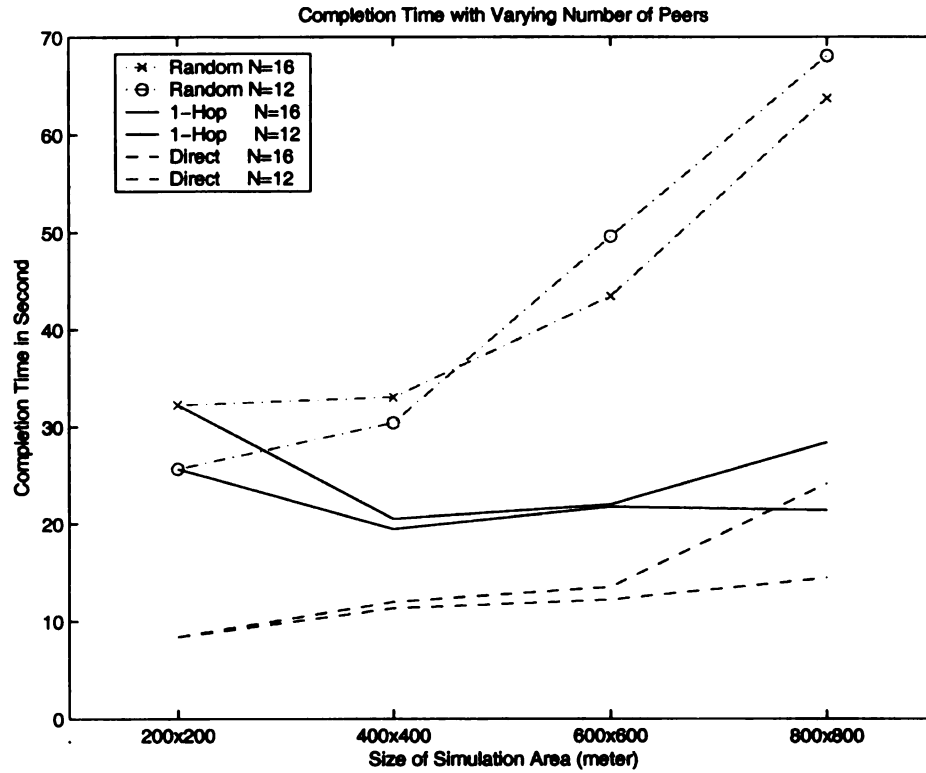


Figure 6.11: Completion time vs. simulation area when N=12 and N=16

Fig. 6.11 shows the completion time of transmitting 1024 byte TCP packets in different size of simulation areas. Each method has two different number of participating peers (N=12 and N=16). In both the 1-hop and the direct methods, the lower line indicates the result when 12 peers are participated, and the upper line for the 16 peer case. The random method shows the crossed lines between 400 meter case and 600 meter case. When all peers are placed within one hop transmission range (200 meter case), the 1-hop method produces the worst performance. It is because only one peer can use the wireless medium at a time in this simulations. In general, as the simulation area enlarges, the completion time also increases, because the number of hops between the source and the destination tends to increase. Fig. 6.12 displays the

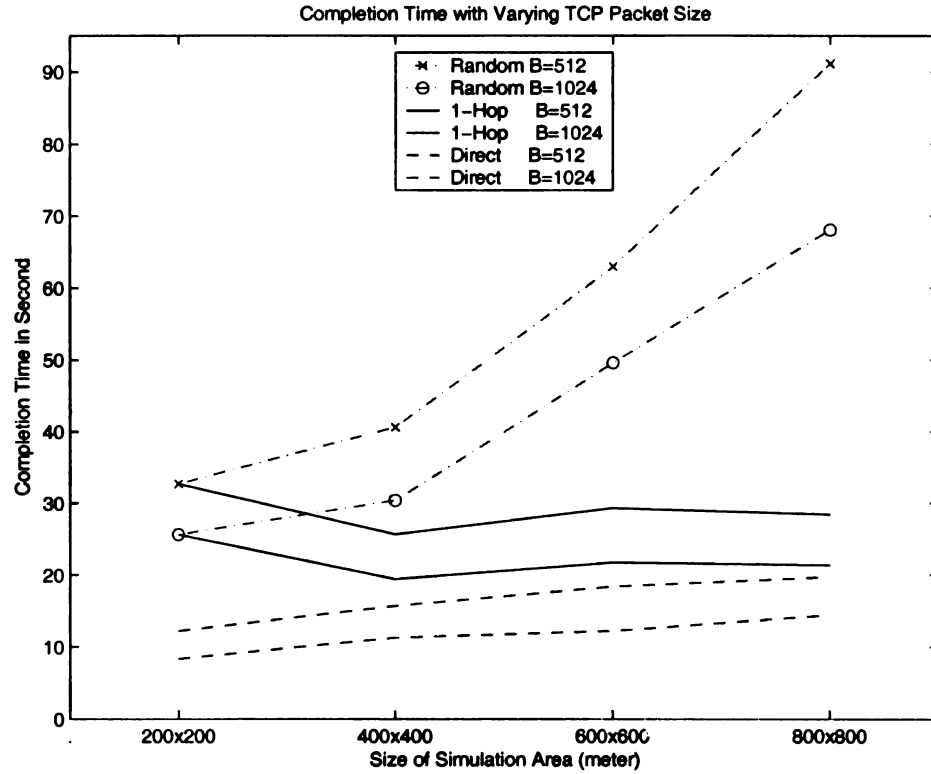


Figure 6.12: Completion time vs. simulation area when packet size is 512B and 1024B completion time in different size of simulation areas when 12 peers are participated. In all methods, the use of small-sized packet takes more time. The 1-hop method takes the longest time in this simulation when all peers are within the transmission range of each other. The 1-hop method shows relatively smaller overhead when using 512 byte of packet than using 1024 byte of packet. In addition, the larger simulation area decreases the overhead of the 1-hop anonymous connection method to 44% (in 800x800 meter case).

6.4 Summary

This chapter introduces the $CHUM_a$ network in which mobile peers reduce telecommunication cost by receiving wanted content (segment) over the cost-free *ad hoc* connection from the peer that has downloaded the content before. When a file is downloaded by a peer, the file may be replicated to several other peers with lower telecommunication cost paid. In an extreme case, some peers may download a file (segment) on behalf of other peers. The sending peer may use a direct *ad hoc* communication path to the receiving peer, whereas the sender may need an anonymous channel to the receiver in order to encourage privacy and to discourage traffic analysis. Two intermediate peers help establish the anonymous connection. Only the associated server of the receiving peer knows both the sender and the receiver of the anonymous connection. No other servers nor any peers do not know about both the actual sender and the receiver. The overhead for the anonymous connection is measured and compared with the direct communication between the sender and the receiver. The simulation results indicate that 1-hop anonymous connection behaves similarly to the direct communication, and the overhead becomes smaller when more peers are participated or the peers are scattered over the larger area.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

This dissertation describes an approach for mobile peers to form an *ad hoc* CHUM network in order to reduce the telecommunication cost by sharing the downloaded data with other peers. Streaming data can be downloaded by one of the participating peers, called proxy, via its WWAN connection, and the proxy chumcasts the data to other peers using its cost-free WLAN connection. In non-streaming data downloading, such as a mobile game program file, each peer downloads a uniquely assigned portion of the file, and exchange it with other peers in order to construct the complete file. In case of streaming data download, all peers take turns being a proxy. One-tier CHUM network operates in a completely distributed manner. The proxy maintains peer membership information and schedules the next proxy. The CHUM network can be implemented by the cooperation between a wired network and a wireless *ad*

hoc network, which is called a two-tier CHUM network. Each peer in a wireless network is associated with a server located in wired networks. All peers upload their neighborhood information periodically to their associated servers. The master server manages peer information, selects rebroadcasting peers, schedules the next proxy, and detects network partitioning. The cost for extra periodic information pays for relieving the burden of computation and communication of peers as well as for reducing the power consumption. In addition, the two-tier cooperation allows the CHUM network to operate in a less vulnerable environment than if it works only in a pure *ad hoc* network. Overall, the CHUM network itself provides peers to save a large amount of the telecommunication costs even if a small number of peers are involved in the network. For example, 80% of the telecommunication cost is saved with a CHUM network size of six to seven peers. Two distribution methods are described for non-streaming data download: per-packet and per-peer based method. The master server performs membership management, download scheduling, and data distribution scheduling. The simulation results show that per-peer based method outperforms per-packet based method. The *ad hoc* CHUM network can be performed as a content storage, and a newly introduced peer may access some content stored in the CHUM network. The source peer may use a shortest *ad hoc* path to the receiving peer, whereas the source may need an anonymous channel in order to maintain privacy and to avoid traffic analysis. Only the associated server of the receiving peer knows both the sender and the receiver of the connection. The overhead of the anonymous connection becomes smaller when large peers are involved and the peers are scattered over the large area.

7.2 Future Work

7.2.1 Buffering in Peers

Buffering is required for peers to download multimedia streaming data. Without buffering, peers may experience gaps while playing the multimedia data. Multimedia traffic is often bursty due to the encoding scheme and the content variation. Smoothing by the content provider may reduce the burstiness of the data traffic. Research suggests that unicast with multiple hops and multicast smoothing by the source reduce both the peak and variability of the bandwidth requirements if buffer spaces are placed at gateway routers or network proxies for the data stream [123, 124]. Peers in a CHUM network perform as proxies and packet forwarders for some time, and the buffer space in peers may reduce the burstiness of multimedia traffic. In addition, the buffer space in peers enables continuous playback of multimedia data even if there are gaps in transmissions to the peers. Optimized buffering scheme may provide for the participating peers to experience smooth playback of multimedia streaming data in a multi-hop wireless *ad hoc* network.

7.2.2 QoS Enhanced Multimedia Downloading

In a multimedia CHUM network, it assumes a single proxy that downloads content from a CP on the Internet. The CP may provide some level of quality of service (QoS) in which additional layer of flow enhances the quality of the content. When all participating peers agree to receive the same level of QoS, multiple proxies may

download their assigned layers in parallel, and distribute them efficiently to all other peers. However, when only some peers want to receive a specific level of QoS, the peers require an additional multicast channel to share the content within them. In addition, an additional proxy scheduling is necessary for the peers that subscribe additional QoS. As a peer subscribes a high level of QoS, it may become a proxy more frequently, because it will be scheduled to be the proxy for both a lower layer and an enhanced layer of the content.

7.2.3 Parallel Downloading with Parallel Distribution

When a CHUM network size is small, such that the largest number of hops between peers is three or four, the per-packet method and the per-peer method perform well. However, when the network size becomes larger and the largest number of hops becomes five or more, two or more peers may transmit their partial content to their neighbors at the same time without collision. Peers can be clustered and a coordinated distribution algorithm controls to exchange the partial content both within the clusters and between the clusters. Each cluster may have at least one gateway peer to exchange content with the gateway peer(s) of other clusters.

7.2.4 Implementation of the CHUM network

The CHUM network is based on the assumption that each peer has two wireless interfaces, one for WWAN for 3G network connection and the other for WLAN that forms an *ad hoc* network. This assumption may be realized in near future, but not

much research work is performed on this assumption. Our future research work includes the implementation of the CHUM network to show the characteristics of the network such as the maximum data receiving rate of a peer depending on the network size, the relationship between the data rate and the buffer size of a peer, and the timing issues like the proper timeout intervals and the service time of a proxy.

BIBLIOGRAPHY

Bibliography

- [1] Lee Garber. Will 3G Really Be the Next Big Wireless Technology? *IEEE Computer*, pages 26–32, Jan 2002.
- [2] IEEE. *IEEE Std 802.11b - Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: High Speed Physical Layer(PHY) in the 2.4 GHz Band*, 1999.
- [3] Pravin Bhagwat. Bluetooth: Technology for Short-Range Wireless Apps. *IEEE Internet Computing*, 5(3):96–103, May-June 2001.
- [4] D. Zhu and M. Mutka. Sharing Presence Information and Message Notification in an Ad Hoc Network. *IEEE Int'l Conference on Pervasive Computing and Communications*, pages 351–358, March 2003.
- [5] Andy Oram. *Peer-to-Peer : Harnessing the Power of Disruptive Technologies*. O'Reilly & Associates, March 2001.
- [6] O. Babaoglu, H. Meling, and A. Montresor. Anthill: a Framework for the Development of Agent-Based Peer-to-Peer Systems. *Proceedings of ICDCS'02*, pages 15–22, July 2002.
- [7] P. Johansson, M. Kazantzidis, R. Kapoor, and M. Gerla. Bluetooth: An Enabler for Personal Area Networks. *IEEE Networks*, 15(5):28–37, Sep-Oct 2001.
- [8] T. Chen, P. Krzyzanowski, M. Lyu, C. Sreenan, and J. Trotter. A VC-based API for Renegotiable QoS in Wireless ATM Networks. *IEEE Int'l Conference on Universal Personal Communication(ICUPC)*, October 1997.
- [9] C. Johnston, P. Narasimhan, and J. Kokudo. Architecture and Implementation of Radio Access Protocol in Wireless ATM Networks. *IEEE Int'l Conference on Communication(ICC)*, June 1998.
- [10] J. Parrow. Verifying a CSMA/CD-Protocol with CCS. *Proceedings of the Eighth IFIP Symposium on Protocol Specification, Testing, and Verification*, pages 373–387, 1988.

- [11] F.A. Tobagi and L. Kleinrock. Packet Seitching in Radio Channels: Part II. The Hidden Terminal Problem in CSMA and busy-tone solution. *IEEE Transactions on Communication*, 23(12):1417–1433, 1975.
- [12] A. Tanenbaum. *Computer Networks, 4th Ed.* Prentice Hall, 2003.
- [13] P. Karn. MACA - A New Channel Access Protocol for Packet Radio. *ARRL/CRRRL Amateur Radio 9th Computer Networking Conference*, September 1990.
- [14] V. Bharghavan, A. Demers, S. Shenker, and L. Zhang. MACAW - A Media Access Protocol for Wireless LAN's. *Proceedings of SIGCOMM '94*, August 1994.
- [15] IEEE. *IEEE Std 802.11 - Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, 1997.
- [16] W. Stallings. IEEE 802.11: Moving Closer to Practical Wireless LANs. *IEEE IT Pro*, May-June 2001.
- [17] IEEE. *IEEE Standard 802.11a - Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: High Speed Physical Layer(PHY) in the 5 GHz Band*, 1999.
- [18] B. McFarland and M. Wong. The Family Dynamics of 802.11. *Queue*, pages 28–38, May 2003.
- [19] C. Heegard. High-Performance Wireless Ethernet. *IEEE Communications Magazine*, pages 64–73, November 2001.
- [20] J. Moy. *OSPF Version 2*. IEEE RFC 2328, April 1998.
- [21] C. Hedrick. *Routing Information Protocol*. IEEE RFC 1058, June 1988.
- [22] J. McQuillan and D. Walden. The ARPA Network Design Decisions. *Compuer Networks*, 1(5):243–289, August 1977.
- [23] J. Jubin and J. Tornow. The DARPA Packet Radio Network Protocols. *Proceedings of the IEEE*, 75(1):21–32, 1987.
- [24] C. Perkins and P. Bhagwat. Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers. *Proc. of ACM SIGCOMM 94*, August 1994.
- [25] C. Chiang, H. Wu, W. Lin, and M. Gerla. Routing in the Clustered Multi-hop, Mobile Wireless Networks with Fading Channel. *Proceedings of IEEE SICON'97*, pages 197–211, April 1997.

- [26] S. Murthy and J. Garcia-Luna-Aceves. An Efficient Routing Protocol for Wireless Networks. *ACM Mobile Networks and Applications Journal, Special Issue on Routing in Mobile Communication Networks*, pages 183–197, October 1996.
- [27] G. Pei, M. Gerla, and T. Chen. Fisheye State Routing: A Routing Scheme for Ad Hoc Wireless Networks. *IEEE ICC 2000*, pages 70–74, June 2000.
- [28] D. Johnson and D. Maltz. Dynamic source routing in ad hoc wireless networks. In *Imielinski and H.Korth, editors, Mobile computing, chapter 5, Kluwer Academic*, 1996.
- [29] C. E. Perkins and E. M. Royer. Ad-hoc On-Demand Distance Vector Routing. *Proc. of the 2nd IEEE Workshop on Mobile Computing Systems and Applications*, Feb 1999.
- [30] V. Park and M. Corson. A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks. *Proceedings of Infocom'97*, April 1997.
- [31] C-K. Toh. Associativity-Based Routing for Ad-Hoc Mobile Networks. *Wireless Personal Communications*, 4(2):1–36, March 1997.
- [32] Y. Ko and N. Vaidya. Location-Aided Routing (LAR) in Mobile Ad Hoc Networks. *Proceedings of ACM/IEEE Mobicom'98*, pages 66–75, October 1998.
- [33] B. Karp and H. Kung. Greedy Perimeter Stateless Routing (GPSR) for Wireless Networks. *Proceedings of ACM/IEEE Mobicom'00*, pages 243–254, August 2000.
- [34] S. Basagni I. Chlamtac, V. Syrotiuk, and B. Woodward. A Distance Routing Effect Algorithm for Mobility (DREAM). *Proceedings of ACM/IEEE Mobicom'98*, pages 76–84, October 1998.
- [35] Z. Haas and M. Pearlman. The Performance of Query Control Schemes for the Zone Routing Protocol. *ACM SIGCOMM'98*, September 1998.
- [36] Z. Haas and M. Pearlman. The Performance of a New Routing Protocol for the Reconfigurable Wireless Networks. *IEEE ICC'98*, June 1998.
- [37] J. Broch, D. Maltz, D. Johnson, Y.-C. Hu, and J. Jetcheva. A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols. *Proc. of ACM/IEEE Mobicom'98*, pages 85–97, October 1998.
- [38] S. Das, C. Perkins, and E. Royer. Performance Comparison of Two On-demand Routing Protocols for Ad Hoc Networks. *IEEE Proceedings of Infocom 2000*, pages 3–12, March 2000.
- [39] S. Deering and D. Cheriton. Multicast Routing in Datagram Internetworks and Extended LANs. *ACM Transactions on Computer Systems*, 8(2):85–110, May 1999.

- [40] T. Ballardie, P. Francis, and J. Crowcroft. Core Based Tree (CBT) - An Architecture for Scalable Inter-Domain Multicast Routing. *Proceedings of ACM SIGCOMM'93*, October 1993.
- [41] J. Moy. Multicast Routing Extensions for OSPF. *Communications of the ACM*, 37(8):61–66, August 1994.
- [42] S. Deering, D. Estrin, D. Farinacci, V. Jacobson, C. Liu, and L. Wei. The PIM Architecture for Wide-Area Multicast Routing. *IEEE/ACM Transactions on Networking*, 4(2):153–162, April 1996.
- [43] S.-J. Lee, M. Gerla, and C.-C. Chiang. On-Demand Multicast Routing Protocol. *Proceedings of IEEE WCNC'99*, Sep 1999.
- [44] E. Royer and C. Perkins. Multicast Operation of the Ad-hoc On-Demand Distance Vector Routing Protocol. *Proceedings of Mobicom'99*, August 1999.
- [45] M. Corson and S. Batsell. A Reservation-Based Multicast (RBM) Routing Protocol for Mobile Networks: Initial Route Construction Phase. *ACM/Baltzer Wireless Networks*, 1(4):427–450, December 1995.
- [46] L. Ji and M. Corson. A Lightweight Adaptive Multicast Algorithm. *Proceedings of IEEE Globecom'98*, pages 1036–1042, November 1998.
- [47] E. Bommaiah, M. Liu, A. MvAuley, and R. Talpade. AMRoute: Ad hoc Multicast Routing Protocol. *Internet Draft, draft-manet-amroute-00.txt (Work in Progress)*, 1998.
- [48] C. Wu and Y. Tay. AMRIS: A Multicast Protocol for Ad hoc Wireless Networks. *Proceedings of IEEE Milcom'99*, November 1999.
- [49] J.J. Garcia-Luna-Aceves and E.L. Madruga. A Multicast Routing Protocol for Ad-Hoc Networks. *Proceedings of IEEE INFOCOM'99*, pages 784–792, Mar 1999.
- [50] J. Garcia-Luna-Aceves and E.L. Madruga. The Core-Assisted Mesh Protocol. *IEEE Journal on Selected Areas in Communications*, 17(8):1380–1394, August 1999.
- [51] E.L. Madruga and J.J. Garcia-Luna-Aceves. Multicasting Along Meshes in Ad-Hoc Networks. *Proceedings of IEEE ICC'99*, pages 314–318, June 1999.
- [52] J. Jetcheva and D. Johnson. Adaptive Demand-Driven Multicast Routing in Multi-Hop Wireless Ad Hoc Networks. *Proceedings of MobiHOC 2001*, October 2001.
- [53] C. Chiang and M. Gerla. On-Demand Multicast in Mobile Wireless Networks. *Proceedings of IEEE ICNP'98*, October 1998.

- [54] T. Ozaki, J. Kim, and T. Suda. Bandwidth Efficient Multicast Routing Protocol for Ad hoc Networks. *Proceedings of IEEE ICCCN'99*, pages 10–17, October 1999.
- [55] P. Sinha, S. Sivakumar, and V. Bharghavan. MCEDAR: Multicast Core Extraction Distributed Ad Hoc Routing. *Proceedings of IEEE WCNC'99*, pages 1313–1317, August 1999.
- [56] S. Lee and C. Kim. Neighbor Supporting Ad hoc Multicast Routing Protocol. *Proceedings of ACM MOBIHOC'00*, pages 37–50, August 2000.
- [57] S. Das, B. Manoj, and C. Murthy. A Dynamic Core Based Multicast Routing Protocol for Ad hoc Wireless Networks. *Proceedings of ACM MOBIHOC'02*, June 2002.
- [58] S.-J. Lee, W. Su, and M. Gerla. Wireless Ad Hoc Multicast Routing with Mobility Prediction. *Mobile Networks and Applications*, 6(4):351–360, 2001.
- [59] B. Williams and T. Camp. Comparison of Broadcasting Techniques for Mobile Ad Hoc Networks. *Proceedings of MOBIHOC'02*, June 2002.
- [60] K. Obraczka, K. Viswanath, and G. Tsudik. Flooding for Reliable Multicast in Multi-Hop Ad Hoc Networks. *Wireless Networks*, 7(6):627–634, November 2001.
- [61] Y. Tseng, S. Ni, Y. Chen, and J. Sheu. The Broadcast Storm Problem in a Mobile Ad Hoc Network. *Wireless Networks*, 8:153–167, March-May 2002.
- [62] Hyojun Lim and Chongkwon Kim. Multicast Tree Construction and Flooding in Wireless Ad Hoc Networks. *ACM MSWiM 2000*, pages 61–68, August 2000.
- [63] J. Wu and F. Dai. Broadcasting in Ad Hoc Networks Based on Self-Pruning. *Proceedings of Infocom 2003*, pages 2240–2250, March/April 2003.
- [64] R. Sivakumar, P. Sinha, and V. Bharghavan. CEDAR: Core Extraction Distributed Ad Hoc Routing. *IEEE Journal on Selected Area in communication*, 17(8):1454–1465, August 1999.
- [65] W. Peng and X. Lu. On the Reduction of Broadcast Redundancy in Mobile Ad Hoc Networks. *Proceedings of MOBIHOC 2000*, Aug 2000.
- [66] A. Qayyum, L. Viennot, and A. Laouiti. Multipoint Relaying for Flooding Broadcast Messages in Mobile Wireless Networks. *Proceedings of Hawaii International System Sciences (HICSS) 2002*, pages 3866–3875, Jan 2002.
- [67] W. Peng and X. Lu. AHBP: An Efficient Broadcast Protocol for Mobile Ad Hoc Networks. *Journal of Science and Technology (JCST) - Beijing, China*, 16(2):114–125, March 2001.

- [68] J. Cartigny and D. Simplot. Border Node Retransmission Based Probabilistic Broadcast Protocols in Ad-hoc Networks. *Telecommunication Systems*, 22(11):189–204, 2003.
- [69] UMTS-Forum. *UMTS-IMT-2000 Spectrum, Report No. 6*, 1999.
- [70] Wireless World Research Forum (WWRF). *WWRF Home Page*. <http://www.wireless-world-research.org>.
- [71] W. Mohr and W. Konhauser. Access Network Evolution Beyond Third Generation Mobile Communications. *IEEE Communications Magazine*, 38:122–133, December 2000.
- [72] M. Metroka. An introduction to narrowband AMPS. *Global Telecommunications Conference (Globecom 1991)*, pages 1463–1468, December 1991.
- [73] S. Hearnden. From TACS to GSM. *IEE National Conference on Telecommunications*, pages 232–238, April 1989.
- [74] Qualcomm. *Qualcomm Home Page*. <http://www.qualcomm.com>.
- [75] M. Mouly and M. Pautet. Current evolution of the GSM systems. *IEEE Personal Communications*, 2(5):9–19, October 1995.
- [76] G. Brasche and B. Walke. Concepts, Services, and Protocols of the New GSM Phase 2+ General Packet Radio Service. *IEEE Communications Magazine*, pages 94–104, August 1997.
- [77] C. Jain and D. Goodman. General Packet Radio Service in GSM. *IEEE Communications Magazine*, pages 122–131, October 1997.
- [78] C. Lindemann and A. Thummier. Performance analysis of the General Packet Radio Service. *In Proceedings of ICDCS'01*, pages 673–680, April 2001.
- [79] C. Lindheimer, S. Mazur, J. Molny, and M. Waleij. Third-generation TDMA. *Ericsson Review* 2, pages 68–79, 2000.
- [80] A. Furuskar, S. Mazur, F. Muller, and H. Olofsson. EDGE: Enhanced Data Rates for GSM and TDMA/136 Evolution. *IEEE Personal Communications*, 6:56–66, June 1999.
- [81] N. Yang. The third Generation Wireless Network Using CDMA Air Interface. *IEEE Wireless Communications and Networking Conference (WCNC)*, pages 649–653, September 1999.
- [82] J. Huber, D. Weiler, and H. Brand. UMTS, The Mobile Multimedia Vision for IMT 2000: a Focus on Standardization. *IEEE Communications Magazine*, pages 129–136, September 2000.

- [83] L. Bos and S. Leroy. Toward an All-IP-based UMTS System Architecture. *IEEE Network*, 15(1):36–45, Jan-Feb 2001.
- [84] K. Richardson. UMTS Overview. *Electronics and Communication Engineering Journal*, 12(3):93–100, June 2000.
- [85] A. Sharma. WCDMA and CDMA2000 compared. *IEEE Wireless Communications and Networking Conference (WCNC)*, 1:23–24, March 2002.
- [86] R. Fisher, H. Suyderhoud, T. Kato, A. Fukasawa, and T. Sato. Wideband CDMA personal communications. *IEEE Vehicular Technology Conference*, 3:1777–1781, May 1997.
- [87] R. Prasad and T. Ojanpera. A survey on CDMA: evolution towards wideband CDMA. *IEEE Proceedings of Spread Spectrum Techniques and Applications*, 1:323–331, September 1998.
- [88] E. Dahlman, B. Gudmundson, M. Nilsson, and A. Skold. UMTS/IMT-2000 based on wideband CDMA. *IEEE Communications Magazine*, 36:70–80, September 1998.
- [89] D. Knisely, S. Kumar, S. Laha, and S. Nanda. Evolution of wireless data services: IS-95 to CDMA2000. *IEEE Communications*, 36:140–149, October 1998.
- [90] Y. Rao and A. Kripalani. CDMA2000 Mobile Radio Access for IMT 2000. *IEEE Int'l Conf. on Personal Wireless Communication*, pages 6–15, February 1999.
- [91] 3GPP. *3GPP Home Page*. <http://www.3gpp.org>.
- [92] W. Lu. 4G mobile research in Asia. *IEEE Communications Magazine*, 41:104–106, March 2003.
- [93] R. Katz. Beyond Third Generation Telecommunications Architectures: The Convergence of Internet Technology and Cellular Telephony. *ACM Mobile Computing and Communications Review*, 2(2):1–5, April 1998.
- [94] M. Frodigh, S. Parkvall, C. Roobol, P. Johansson, and P. Larsson. Future-Generation Wireless Networks. *IEEE Personal Communications*, 8(5):10–17, October 2001.
- [95] J. Craninckx and S. Donnay. 4G terminals: how are we going to design them? *ACM/IEEE Design Automation Conference*, pages 79–84, June 2003.
- [96] Bluetooth. *Official Bluetooth Consortium Site*. <http://www.bluetooth.org>.
- [97] J. Haartsen. The Bluetooth Radio System. *IEEE Personal Communications*, February 2000.

- [98] B. Chatschik. An Overview of the Bluetooth Wireless Technology. *IEEE Communications Magazine*, 39:86–94, December 2001.
- [99] Maria Papadopouli and Henning Schulzrinne. Connection Sharing in an Ad Hoc Wireless Network among Collaborating Hosts. *NOSSDAV 1999*, June 1999.
- [100] D. Zhu and M. Mutka. Fair Sharing of Proxy Responsibilities in an Ad Hoc Network. *IEEE Int'l Conference on Pervasive Computing and Communications*, March 2004.
- [101] Gnutella. *The Gnutella Home Page*. <http://www.gnutella.wego.com>.
- [102] Jungle Monkey. *The Jungle Monkey Home Page*. <http://www.junglemonkey.net>.
- [103] P. Druschel and A. Rowstron. PAST: A Large-Scale Persistent Peer-to-Peer Storage utility. In *Proceedings of Workshop on Hot Topics in Operating Systems(HotOS-VIII)*, May 2001.
- [104] I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan. Chord: A Scalable Content-Addressable Network. In *Proceedings of the ACM SIGCOMM 2001 Technical Conference*, August 2001.
- [105] I. Clark, O. Sanberg, B. Wiley, and T. W. Hong. Freenet: A Distributed Anonymous Information Storage and Retrieval System. In *Proceedings of Workshop on Design Issues in Anonymity and Unobservability*, <http://freenet.sourceforge.net>, July 2000.
- [106] A. Rubin, M. Waldman, and L. Cranor. Publius: A robust, Tamper-evident, Censorship Resistant, Web Publishing System. In *Proceedings of the 9th USENIX Security Symposium*, August 2000.
- [107] Mojo Nation. *The Mojo Nation Home Page*. <http://www.mojonation.net>.
- [108] R. Dingledine, D. Molnar, and M. J. Freeman. The Free Haven Project: Distributed anonymous storage service. In *Proceedings of the Workshop on Design Issues in Anonymity and Unobservability*, July 2000.
- [109] Y. Chu, S. Rao, S. Seshan, and H. Zhang. A Case for End System Multicast. *IEEE Journal on Selected Areas in Communications(JSAC)*, 20(8), October 2002.
- [110] J. Apostolopoulos, T. Wong, W. Tan, and S. Wee. On Multiple Description Streaming with Content Delivery Networks. In *Proceedings of INFOCOM 2002*, pages 1736–1745, June 2002.
- [111] A. Basso, C. Cranor, R. Gopalakrishnan, M. Green, C. Kalmanek, D. Shur, S. Sibal, C. Sreenan, and J. Merwe. PRISM: an IP-based Architecture for Broadband Access to TV and other Streaming Media. In *Proceedings of IEEE*

International Workshop on Network and Operating System support for Digital Audio and Video, June 2000.

- [112] J. Jannotti, D. Gifford, K. Johnson, M. Kaashoek, and J. O'Toole. Overcast: Reliable Multicasting with an Overlay Network. In *Proceedings of USENIX OSDI 2000*, October 2000.
- [113] D. Tran, K. Hua, and T. Do. ZIGZAG: An Efficient Peer-to-Peer Scheme for Media Streaming. In *Proceedings of INFOCOM 2003*, Mar 2003.
- [114] D. Xu, M. Hefeeda, S. Hambruch, and B. Bhargava. On Peer-to-Peer Media Streaming. *ICDCS 2002*, July 2002.
- [115] M. Castro, P. Druschel, A. Kermarrec, A. Nandi, A. Rowstron, and A. Singh. SplitStream: High-Bandwidth Multicast in Cooperative Environments. *ACM Symposium on Operating Systems Principles (SOSP'03)*, pages 298–313, October 2003.
- [116] BitTorrent. *The BitTorrent file distribution system*. <http://bitconjurer.org/BitTorrent/>.
- [117] K. Egevang and P. Francis. *The IP Network Address Translator (NAT)*. IEEE RFC 1631, May 1994.
- [118] H. Zhou, L. Ni, and M. Mutka. Prophet Address Allocation for Large Scale MANETs. In *Proceedings of INFOCOM 2003*, Mar 2003.
- [119] G. Calinescu, I. Mandoiu, P. Wan, and A. Zelikovsky. Selecting Forwarding Neighbors in Wireless Ad Hoc Networks. *ACM Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, pages 34–43, July 2001.
- [120] Sudipto Guha and Samir Khuller. Approximation Algorithms for Connected Dominating Sets. *Algorithmica*, 20(4):374–387, 1998.
- [121] Yih-Chun Hu, Adrian Perrig, and David B. Johnson. Ariadne: A Secure On-Demand Routing Protocol for Ad Hoc Networks. *Proceedings of MOBICOM'02*, September 2002.
- [122] Hongmei Deng, Wei Li, and Dharma P. Agrawal. Routing Security in Wireless Ad Hoc Networks. *IEEE Communications Magazine*, pages 70–75, October 2002.
- [123] Jenifer Rexford and Don Towsley. Smoothing Variable-Bit-Rate Video in an Internetwork. *IEEE/ACM Transactions on Networking*, 7(2), April 1999.
- [124] S. Sen, D. Towsley, Z. Zhang, and J. Dey. Optimal Multicast Smoothing of Streaming Video over an Internetwork. In *Proceedings of INFOCOM 1999*, March 1999.

- [125] Arun Solleti and Kenneth Christensen. Efficient Transmission of Stored Video for Improved Management of Network Bandwidth. *International Journal of Network Management*, 10:277–288, 2000.
- [126] S. Kang and M. Mutka. Efficient Mobile Access to Internet Data via a Wireless Peer-to-Peer Network. *IEEE Int’l Conference on Pervasive Computing and Communications*, pages 197–205, March 2004.
- [127] National Bureau of Standards. *Data Encryption Standard*. U.S. Department of Commerce, FIPS, pub. 46, January 1977.
- [128] NIST. *Advanced Encryption Standard (AES)*. <http://csrc.nist.gov/CryptoToolkit/aes/>.
- [129] R. Rivest. *The MD5 Message-Digest Algorithm*. IEEE RFC 1321, April 1992.
- [130] ns2. *The network simulator*. <http://www.isi.edu/nsnam/ns>.
- [131] M. Reed, P. Syberson, and D. Goldschlag. Anonymous Connections and Onion Routing. *IEEE Journal on Selected Areas in Communications*, 16(4):482–494, May 1998.

MICHIGAN STATE UNIVERSITY LIBRARIES



3 1293 02504 3930