# Studies on Data Transfer with Concurrent and Integrated Use of Multiple Networks

Akira NAGATA

# Preface

The use of smartphone devices running high-functionality applications is widespread among the general public. Application data are often stored at a data center in a cloud computing network, meaning that many applications rely on data communication through wide area networks including wireless access networks and wired backbone networks. This results in an increase in demand for the transfer of reliable and large data sets at all times and in all places — including mobile environments. However, it is, essentially, difficult for applications to transfer data in a sufficiently stable manner and at a fast enough rate over a single wireless network.

Fortunately, there are many wireless networks offering various options because wireless broadband services — such as cellular 3G, LTE, 802.11 wireless LAN, and WiMAX — have become widely deployed in many areas. Therefore, we believe it is important to explore how to exploit these multiple and heterogeneous available networks in order to obtain the performance required to meet the requirements of applications with respect to data transfer.

Although numerous studies have focused on the selection of a communication medium among multiple alternative networks with the aim of maintaining connectivity, or on the concurrent use of multiple networks to aggregate bandwidth, none have focused on how to exploit multiple networks on the basis of the characteristics, status and requirements of both application and network. With this in mind, we tackled the issue of data transfer over a wide area network with the goal of achieving quick, efficient and reliable data transfer by exploiting the available resources of wired and wireless networks. More specifically, we focused

on a data-transfer framework to achieve concurrent and integrated use of multiple and heterogeneous wireless access networks or multiple routes in a single network on the basis of characteristics, environmental situation, and status, as well as the application requirement.

We begin our study by addressing the question of how to quickly transfer a large file if multiple and heterogeneous networks are available but when none of them individually has sufficient performance for the requested task. In particular, in the case of one-to-one data transfer between a server in a wired backbone network and a mobile host in a wireless access environment, where high-speed data communication is not always available, we discuss and propose a framework of data transfer through the concurrent and integrated use of multiple and heterogeneous networks. In this framework, a control information flow for reliable end-to-end transport is handled separately from data flow on different network paths in different ways.

We then consider how to assist rapid and efficient data transfer from a server in a wired backbone network to a mobile host in a wireless access environment, we discuss and propose a one-to-many file-delivery scheduling scheme for a server that will quickly and efficiently send a file to multiple wireless access base stations by the concurrent use of multiple available routes. We implemented a proof-of-concept model for our proposals and conducted experiments to evaluate performance and clarify any practical issues to be addressed. Through our research, we demonstrated the potential for a data-transfer framework with concurrent and integrated use of multiple and heterogeneous networks and have contributed to its practical implementation. This dissertation is organized as follows.

First, in Chapter 1, we present the requirements and problems to be solved with respect to achieving reliable and efficient data transfer, especially in a wireless access network that does not have adequate performance.

Second, in Chapter 2, we present a survey of related work and the background to this research, which includes data transfer using multiple networks, data transfer when using a network with inadequate performance, and route planning for one-to-many data transfer to

maximize bandwidth.

Third, in Chapter 3, we introduce the concept of integrating multipath data transfer (IMPDT) to efficiently transfer a large packet of data, which integrates heterogeneous networks not simply for bandwidth aggregation but also to provide sustainable control information exchange on a stable but low data rate network path, being handled separately from data transmission on different network paths in different ways. We also discuss the design of a proof-of-concept implementation for an IMPDT framework.

Fourth, in Chapter 4, we discuss three field experiments conducted on an IMPDT prototype system using a combination of terrestrial communication links and different satellite communication links to validate our prototype implementation in a real-world communication services environment. Through these experiments, we proved that our prototype implementation performed better than a simple aggregation of transfer by parallel TCP connections in unstable wireless network environments, and that our system worked as expected—even in a moving vehicle environment—due to a slow but stable satellite link for control information flow. We also verified that our system exploited its characteristic of a high-speed satellite link even under lossy and/or disrupted conditions by complementary use of a terrestrial 3G network. We also identified some problems in a real-world communication service environment and improved our system.

Fifth, in Chapter 5, we discuss the efficient and practical use of IMPDT for uses other than a simple bulk file transfer. We designed and implemented a prototype web access system for web browsing over multiple wireless networks by integrating the HTTP protocol and the IMPDT scheme. Our system handles not only control and data flows for end-to-end transport differently, but also for each file, which consists of a web page, and selects an appropriate transport-layer method for the file.

Sixth, in Chapter 6, we propose a multipath multicast (MPMC) file-transfer scheme that contributes to reduce transmission times for one-to-many file transfers from a server to multiple wireless access base stations. This contribution was realized by: (i) fully utilizing the

possible network bandwidth by transmitting segments of a file simultaneously on multiple paths, and, (ii) reducing the bandwidth required to deliver a file to multiple receivers by multicasting. We developed a prototype MPMC system for file transfers on OpenFlow networks and experimentally examined its feasibility and usefulness in three scenarios.

In closing, I would like to say that I hope this dissertation will be helpful for further study in this field.

December 2013

Akira NAGATA

# Acknowledgements

First of all, I wish to express my sincere appreciation to Professor Masato Tsuru at Kyushu Institute of Technology. His constant encouragement, guidance through this research, invaluable discussions and advices have greatly helped in accomplishing the research. I also thank him for his careful reading of all papers on the research and assignment to research and development projects.

I would like to express my gratitude to Professor Akihiro Fujiwara, Kenji Kawahara, Eiji Miyano at Kyushu Institute of Technology, and Professor Takahiro Matsuda at Osaka University for their invaluable discussion, comments, and careful reading of my papers and their useful advices on the research.

I wish to thank Professor Yuji Oie at Kyushu Institute of Technology for his invaluable discussion, comments, and kindly supports any time.

I wish to thank Professor Tetsuya Takine at Osaka University, Professor Hiroyoshi Miwa at Kwansei Gakuin University, and Professor Masato Uchida at Chiba Institute of Technology for their invaluable discussion, comments, useful advices, and collaboration on the research.

I also wish to thank Shinya Yamamura at Fujitsu Kyushu Network Technologies for his invaluable discussion and cooperation conducting the field experiment.

Finally, my greatest appreciation goes to our company president, Hiroshi Takagi , and our company director, Katsuichi Nakamura, for their understandings and support for me. And, I extent thank to our company members, and all other friends for their warm encouragement

and supports.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

A number of wireless broadband services have been being developed in response to the demand for high-speed data communications anytime, anyplace—even in a mobile environment. For example, 802.11 wireless LAN hot-spots are now widespread, and LTE (long term evolution) and WiMAX (worldwide interoperability for microwave access) services have been established in many urban areas of Japan. High-speed satellite communications have also been developed.

Recent demands on mobile wireless communications have prompted research into the best ways of selecting multiple wireless communication media depending on the time and the place. Within wired access and core networks, traffic flow distribution (by traffic engineering) over multiple routes has been introduced to increase the efficiency of network link usage and the quality of service (QoS) of each individual traffic flow. The use of multiple network paths between end application nodes has attracted a great deal of attention recently as a way to increase application performance and reliability.

However, due to the high deployment costs and the nature of radio communications, providing wireless broadband services anytime and anyplace with a sufficiently high performance is a difficult task. This presents a challenge to the use of wireless broadband services, and end-to-end data transfer cannot perform efficiently. In cases where an individual net-

work cannot provide the communication quality required for application data transfer, multiple networks should be utilized concurrently to complement each other so as to provide an acceptable quality of communication. However, it is difficult to realize optimum integration of multiple network paths because the Internet was not originally designed to utilize multiple networks.

This chapter is organized as follows. In Section 1.1, we introduce the requirements of and the problems facing reliable and efficient data transfer, especially in a challenged wireless network that does not have adequate performance. Next, in Section 1.2, we present the challenges involved in data transfer with concurrent and integrated use of multiple and heterogeneous networks. Finally, we show the outline of the dissertation in Section 1.3.

## 1.1 Requirements and Problems Facing Data Transfer in Challenged Networks

The use of smartphone devices that operate high-functionality applications is widespread among the general public. Application data are often stored at a data center in a cloud computing network, meaning that many applications rely on data communications. This results in an increase in demand for the transfer of large data sets at all times and in all places—including mobile environments. This requirement is currently realized by the widespread dissemination of high-speed broadband wireless access networks (i.e., ubiquitous networking).

However, as stated earlier, a wireless network cannot, inherently, avoid becoming challenged while trying to provide the communication quality required by an application. A challenged network suffers from long delays, heavy packet losses, and frequent disconnections. Therefore, data-transfer technology should not focus solely on providing high-speed, stable network environments, but should be extended to work efficiently—even in a challenged network environment.

With respect to reliable data transfer, the Internet uses TCP as data-transfer technology. But in a challenged network situation, end-to-end data transfer by conventional TCP does not perform efficiently. This is because it is assumed that data and control information are either conveyed in the same packet or, if they are in different packets, by the same network path (communication medium). When a network path is transiently disconnected, not only data—but also the associated control information—are blocked, which can result in a significant degradation in performance of reliable data transfer. The exchange of information essential for the preparation of application data exchange, such as address resolution and authentication, may also fail in an unstable network path.

It should be noted that a wireless channel that covers a larger area generally has a lower transmission rate, and thus such channels are not suitable for the transfer of large packets of data due to the narrow bandwidth of the channel. However, wireless channels can prove useful for the stable exchange of small packets of data, such as the control information required for file transfer.

Neither network selection from among multiple available networks or naive simultaneous utilization of multiple available networks provides a good solution for efficient data transfer when an individual network cannot provide the communication quality required of application data transfer. In other words, multiple networks should be utilized concurrently to complement each other so as to provide an acceptable quality of communication. In order to realize efficient data transfer in a challenged network environment, our research goal is to construct a novel data-transfer framework that enables the concurrent and integrated use of heterogeneous multiple networks.

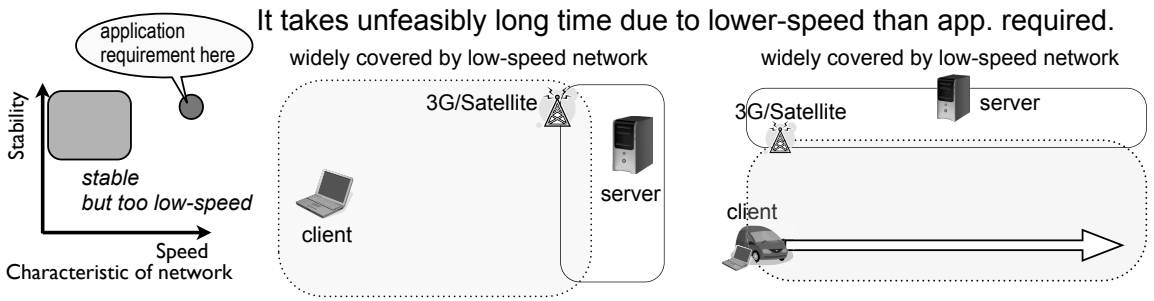


Figure 1.1: Challenged network example (stable, but too slow for applications)

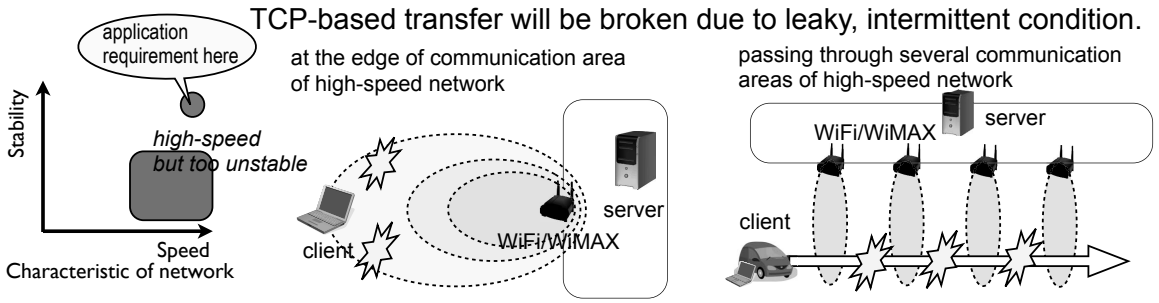


Figure 1.2: Challenged network example (fast, but too unstable for applications)

## 1.2 Challenges in Concurrent and Integrated Use of Multiple Networks

There are many wireless networks of different types available because wireless broadband services—such as cellular 3G, LTE, 802.11 wireless LAN, or WiMAX—have been widely deployed in many areas. Mobile devices also contain plural communication interfaces of various types. Therefore applications on mobile devices are able to access multiple wireless networks of different types at the same time. With respect to the requirements of an application, some available wireless networks might be fast enough but too unstable, as shown in Fig. 1.2. Another might be sufficiently stable but too slow, as shown in Fig 1.1. If a

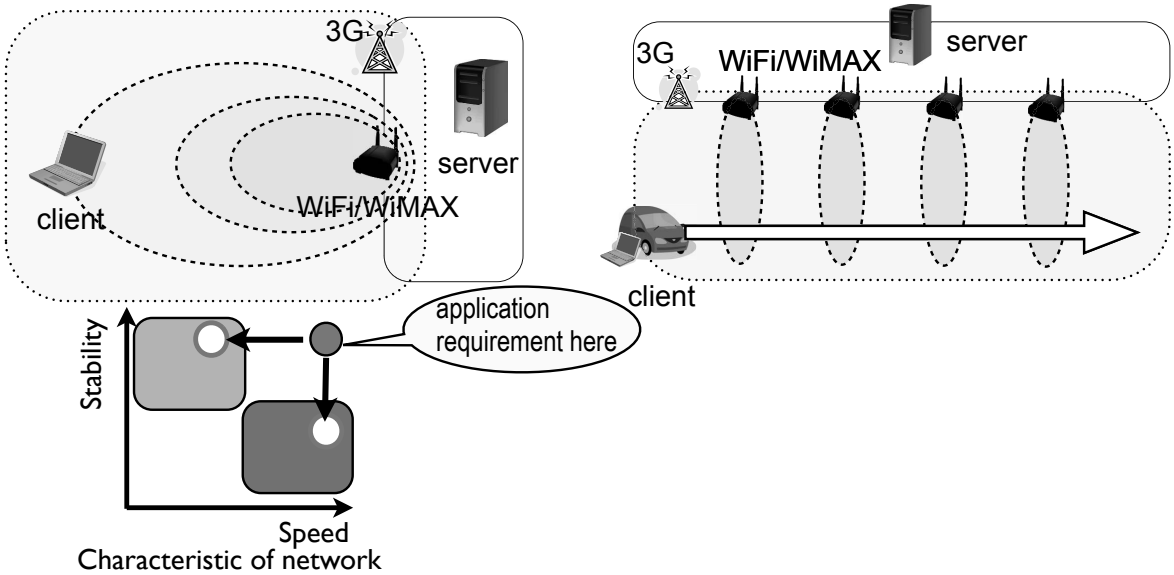


Figure 1.3: Approach: Integration of both types of challenged networks

mobile device can concurrently use both types of networks, it could handle them differently based on their characteristics and have them complement each other to provide an acceptable quality of communication (as shown in Fig. 1.3). This is the fundamental motivation of our research.

We propose a new framework that can rapidly transfer large packets of data between a server in a wired backbone network and a mobile host in a wireless access network by integrating multiple and heterogeneous challenged wireless access networks, e.g., unstable networks with a high data rate and stable networks with a low data rate [1, 2, 3, 4]. Our basic idea is simple; separate the control information flow from the data flow. The control information flow does not need to travel over the same communication medium as the data flow. Instead, a control flow via a more sustainable medium covering a wider area would result in a steady and therefore timely exchange of control information, even if the data flow were dynamically disrupted.

With regard to one-to-one data transfer between a server in a wired backbone network

and a mobile host in a wireless access network, we discuss and design its functions, focusing on the following points:

- separate each data to be transferred on the basis of the performance category that an application requests
- handle each data differently according to the requested performance (including conveying it through a network with a specific characteristic, or process it so as to achieve its requested performance if necessary (e.g., duplication or redundant encoding))
- detect available networks toward the destination and notify the network to be used
- retransmit any lost packets to provide reliable data transfer in an efficient manner
- control sending rate to make the best use of network resources and to avoid suppressing other traffic that share the network

Additionally, the framework should be applicable to existing networks and be practical to use. We discuss how to apply our proposal not only to a simple bulk-data transfer but also to an interactive application [5, 6]. It is not always easy to use an interactive application in challenged networking environments, because continuous connectivity during use is assumed.

From the perspective of quick and efficient data transfer from a server in a wired backbone network to a mobile host in a wireless access environment, we discuss and propose a one-to-many file-delivery scheduling scheme to enable a server to quickly and efficiently send a file to multiple wireless access base stations by the concurrent use of multiple available routes [7]. The goal is to decrease file-transfer time by aggregating the bandwidth of each sender-receiver path and by multicast transfer for efficient bandwidth utilization.

## 1.3 Outline of this dissertation

In Chapter 2, we present a survey of related work and the background to this research, which includes data transfer using multiple networks, data transfer when using a network with inadequate performance, and route planning for one-to-many data transfer to maximize bandwidth.

Third, in Chapter 3, we introduce the concept of integrating multipath data transfer (IMPDT) to efficiently transfer a large packet of data, which integrates heterogeneous networks not simply for bandwidth aggregation but also to provide sustainable control information exchange on a stable but low data rate network path, being handled separately from data transmission on different network paths in different ways. We also discuss the design of a proof-of-concept implementation for an IMPDT framework.

Fourth, in Chapter 4, we discuss three field experiments conducted on an IMPDT prototype system using a combination of terrestrial communication links and different satellite communication links to validate our prototype implementation in a real-world communication services environment. The first experiment uses a combination of 3G, Wi-Fi, and a satellite communication link (ETS-VIII) which is regarded as a slow but stable network and better suited for control flow in this experiment. In the second experiment, a commercial global satellite communication network is used as a stable but low data rate link for emergency networking in disaster fields. The third experiment uses a pair of a high-speed satellite communication links: (WINDS) and 3G. The WINDS link is regarded as having a high data rate but being transiently lossy and a disrupted link in this experiment.

Through these experiments, we discuss the performance of our prototype implementation compared to a simple aggregation of transfer by parallel TCP connections in unstable wireless network environments, and the experimental results in a moving vehicle environment. We also verify if our system exploits its characteristic of a high-speed satellite link even under lossy and/or disrupted conditions by complementary use of a terrestrial 3G network.

Fifth, in Chapter 5, we discuss the efficient and practical use of IMPDT for uses other

than a simple bulk file transfer. We designed and implemented a prototype web access system for web browsing over multiple wireless networks by integrating the HTTP protocol and the IMPDT scheme. Our system handles not only control and data flows for end-to-end transport differently, but also for each file, which consists of a web page, and selects an appropriate transport-layer method for the file.

Sixth, in Chapter 6, we propose a multipath multicast (MPMC) file-transfer scheme that contributes to reduce transmission times for one-to-many file transfers from a server to multiple wireless access base stations. This contribution was realized by: (i) fully utilizing the possible network bandwidth by transmitting segments of a file simultaneously on multiple paths, and, (ii) reducing the bandwidth required to deliver a file to multiple receivers by multicasting. We discuss the design of a prototype MPMC system for file transfers on OpenFlow networks and the results of experiments to examine its feasibility and usefulness in three scenarios.

# Chapter 2

# Related Work

In conventional approaches to the utilization of multiple networks, a communication medium is selected either manually by a user or dynamically by horizontal or vertical handover[8, 9, 10, 11]. New protocols such as the stream control transmission protocol (SCTP) [12] and the datagram congestion control protocol (DCCP) [13] enable more than one media to be simultaneously used in multihoming environments. An extension of SCTP [14] has been proposed to overcome performance problems arising from re-ordering. Another study successfully extended TCP-like congestion control for multiple end-to-end paths [15]. However, if none of the available network paths are stable with a high data rate, it takes a long time to complete the transfer of a large data packet and communications may fail due to frequent timeouts, irrespective of the communication media selected.

Delay, disruption, and/or disconnection tolerant networking (DTN) is an emerging research area for addressing problems resulting from the characteristics of challenged networks such as long delays, heavy packet losses, and frequent disconnections, in which the conventional TCP/IP-based model does not work well [16, 17]. LTP-T [18], which is a multi-hop extension of LTP [19], is a retransmission-based reliable transport protocol tolerant of end-to-end long delays and frequent disconnections. It cannot, however, make the best use of multiple network paths in the scenarios assumed in this paper.

Pradeep Kyasanur et al. [20] proposed a media access control (MAC) protocol for transmitting a layer-2 control message (RTS or CTS) and layer-2 data (user data or ACK) in different frequency bands simultaneously, which improves the throughput of 802.11 wireless LANs. To exchange control information for routing from nodes other than a neighbor, the resource allocation protocol for intentional DTN (RAPID) [21] refers to the usage of an external network. N. Banerjee et al. [22] proposed simultaneous usage of multiple wireless communication media installed within the same system: Wi-Fi and Xtend (900 MHz radio), which has a longer range than Wi-Fi, detects a Wi-Fi client in a vehicle moving into the area of a Wi-Fi hot-spot, and awakens the Wi-Fi access point from sleep mode before the client nears the point.

For a large data transfer, especially in error-prone challenged networks, there are two approaches—reactive or proactive. A simple duplication is proactive, and transfer with redundancy by forward error/erasure coding (FEC) is a more sophisticated approach [23]. This could flexibly utilize as many resources in a leaky network as possible [24]. In proactive approaches, however, the efficient use of network resources is not easy because of excess redundancy or duplicate data in general, especially when multiple users share challenged networks with limited resources. A reactive approach using NACK-based retransmission is expected to suppress excess data transfer by timely feedback, which minimizes resource (bandwidth) consumption. This kind of reactive retransmission is basically in a class of well-known methods such as selective repeat ARQ (automatic repeat-request) [25] as a way of enabling a sender to retransmit lost data.

With regards to efficient bandwidth utilization over multiple routes, the maximum flow problem has been studied extensively for one-to-one data transfer. Solutions to such problems realize the maximally feasible data-rate (maxflow) from a single sender to a single receiver. All possible link capacities are exploited simultaneously by using multiple network routes over relay nodes (paths) from sender to receiver. When a sender host sends a file to multiple receiver hosts, it is generally expected that in a reliable multicast (MC) file transfer

(one-to-many), the traffic load is less than that of multiple unicast file transfers (one-to-one). In general, multicasting along a tree over relay nodes from a single sender to multiple receivers is efficient in terms of link capacity consumption. However, the data rate is limited by the weakest receiver in the MC tree. This problem occurs, for example, in the case of the heterogeneity of the link capacity over which an MC flow traverses or the processing capacity of receiver hosts within the same MC groups. To address this heterogeneity problem, multicasting using layered coding [26, 27] enables heterogeneous receiver hosts to receive MC flows at an appropriate rate. In Ref. [28], to achieve non-real-time point-to-multipoint content delivery services over heterogeneous environments, a transit node temporarily stores data streams to bridge the gap between the incoming receiving rate from an upstream node and the outgoing available sending rate to a downstream node. The fountain code-based multipath transmission control protocol [29] employs a fountain code to encode transmission data and achieves higher reliability without retransmissions. Although multiple MC data transfer with network coding achieves maxflow [30, 31], there remain many challenges to be addressed, such as a way to realize synchronization of packets to be coded and ways to reduce combination and coding packet overhead.

# Chapter 3

# Data Transfer Exploiting Multiple Heterogeneous Challenged Networks

## 3.1 Concept of Concurrent and Integated Use

We briefly review the concept of file transfers over multiple network paths (communication media). Suppose long delays, leaks (heavy packet losses) and frequent disconnection are observed in these paths; hence, this is a challenged environment.

We consider the situation that multiple and heterogeneous networks are simultaneously available. At least one is a stable network, even if its speed is too slow for a given file to be transferred within a reasonable time. Here, "*stable*" means continuously available for a longer time and in a wider area. Other networks have some challenged characteristics even if their speed is fast. Figure 3.1 shows a conceptual example of the proposed system called integrating multi-path data transfer (IMPDT). Assume that three types of communication media ($NW_1$, $NW_2$, and $NW_3$) are available to transfer large data sets (hundreds of Mbytes). $NW_1$ is a stable network with a low data rate (e.g. dozens of Kbps). $NW_2$ and $NW_3$ have sufficiently high data rates but are not sufficiently stable; the probability of loss is high (e.g., more than 5–10%), and the networks frequently drop connections.

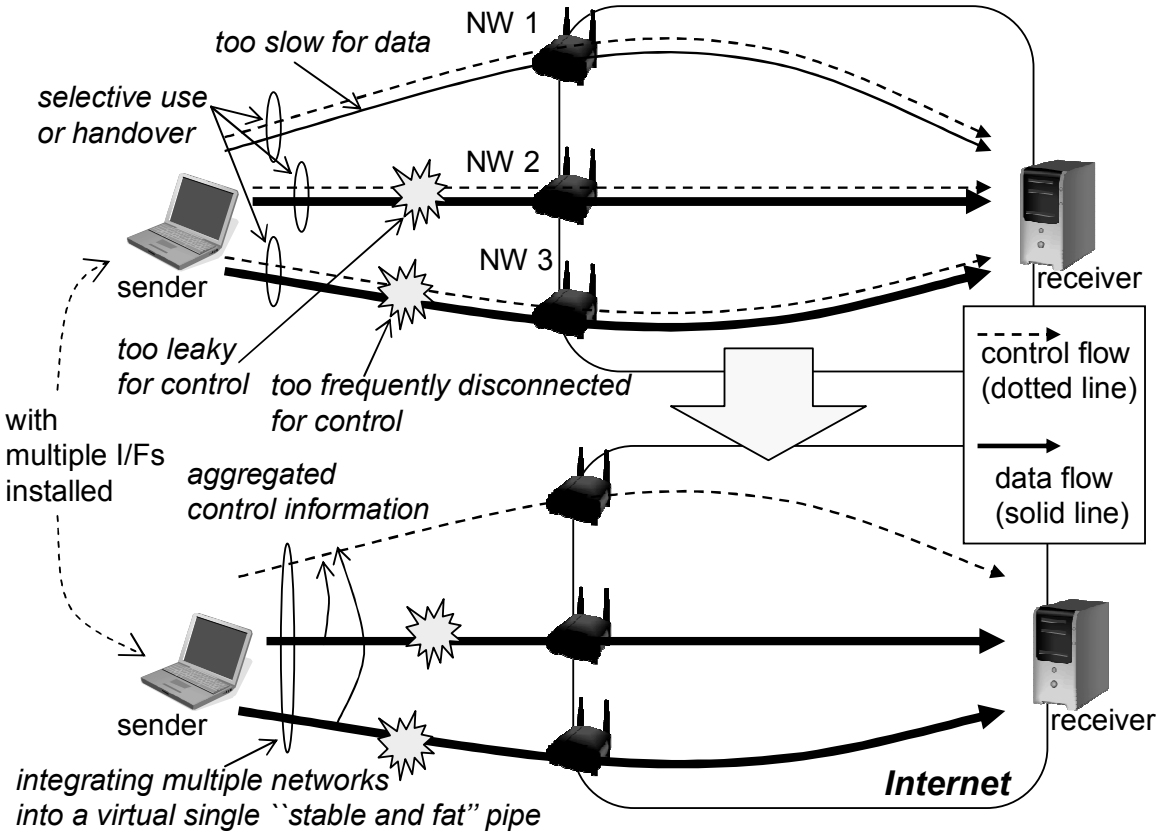


Figure 3.1: Proposed concept

The proposed system takes advantage of integration of multiple, heterogeneous and diverse networks, not simply for bandwidth aggregation but also particularly for providing stable control information exchange for end-to-end reliable transfer. In the proposed approach, file transfer is performed by multiple separated communication flows of control information for reliable and efficient data transmission (control flow), and those of the data transmission itself (data flow). Each flow is distinguishably transmitted and can be conveyed by different communication media, depending upon the characteristics of the flow and communication medium. A control flow conveys information for retransmission (ACK or NACK), rate adjustment, and so on. The size of control flow is relatively much smaller than that of the

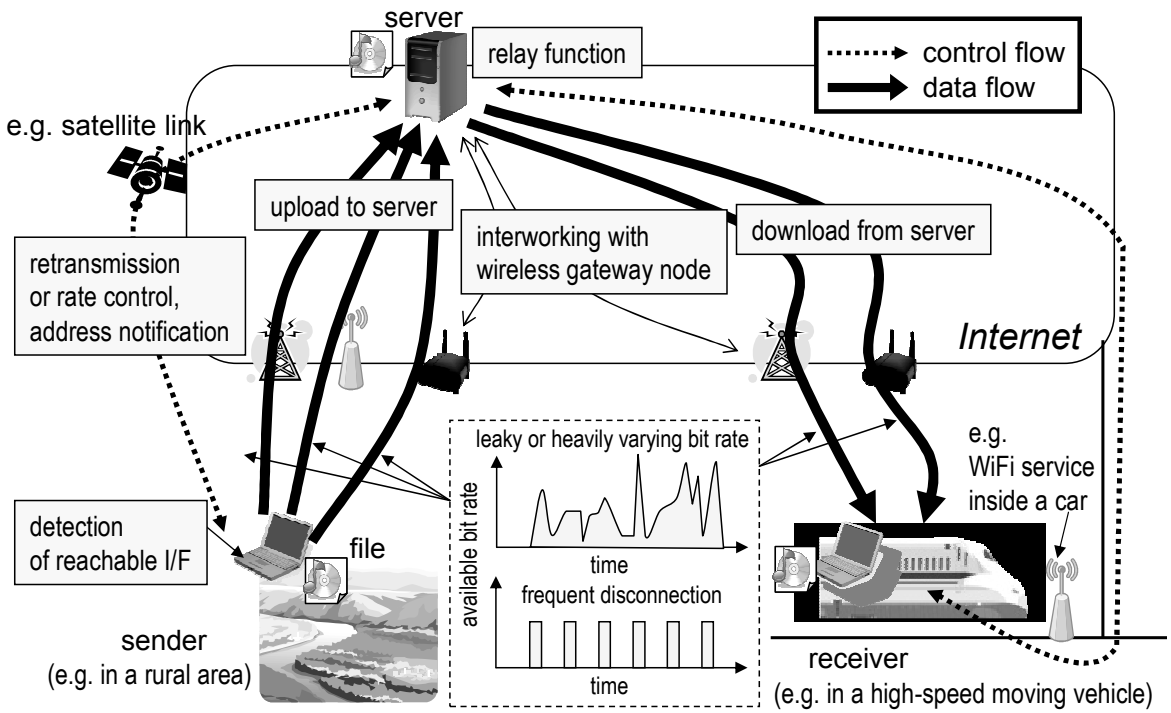


Figure 3.2: An overview of our proposal

data, so the bandwidth required for network to convey it can be very small. Instead, it needs stable transfer. In the proposed system, control flow may not necessarily be through the same communication media as data flows; it may reach the receiver through more stable communication medium that also covers a wider area. For example in Figure 3.1, steady and consequently timely exchange of control information via a stable network such as $NW_1$ enables data transfer in an unstable or intermittent network such as $NW_2$ or $NW_3$.

The proposed system supports both senders (as shown in Figure 3.1) and receivers in challenged networks. Using the control flow, the receiver notifies the sender host of the preferred interfaces that may receive data flows. When both senders and receivers are in a challenged network, a relay node in the Internet, as shown in Figure 3.2, is required for the efficient transfer of large data sets.

Two example scenarios are shown in Figures 3.3 and 3.4. In Figure 3.3, the client on

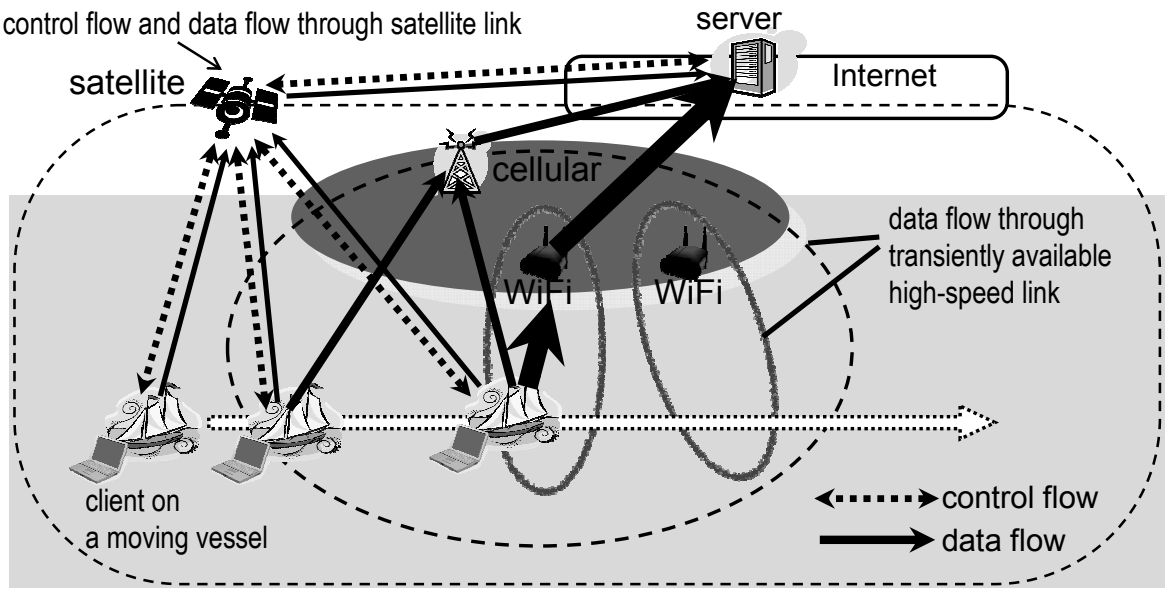


Figure 3.3: Use case example of IMPDT (1)

a moving vessel wants to upload or download a large file from/to a server on the Internet while traversing several broadband hot-spots such as Wi-Fi or WiMAX that suffers from disconnection and disruption. A situation we assume in Figure 3.3 is as follows. A satellite communication link makes a client and a server exchange a file anytime, but its data rate is not high enough for the size of the file to be transferred. On the other hand, terrestrial communication links such as 3G, Wi-MAX or Wi-Fi are only transiently available between a client and a server during the file transfer, but the data rates are higher compared to the satellite link. These broadband but transient terrestrial links can be used concurrently with the sustainable satellite link to add the data rate temporarily.

By contrast, in Figure 3.4, a client and a server transfer a file mainly through a satellite communication link, of which data rate is high enough for the file to be transferred, while it is transiently interrupted or disconnected during the file transfer due to environmental changes. Another terrestrial communication link such as 3G is also available between a client and a

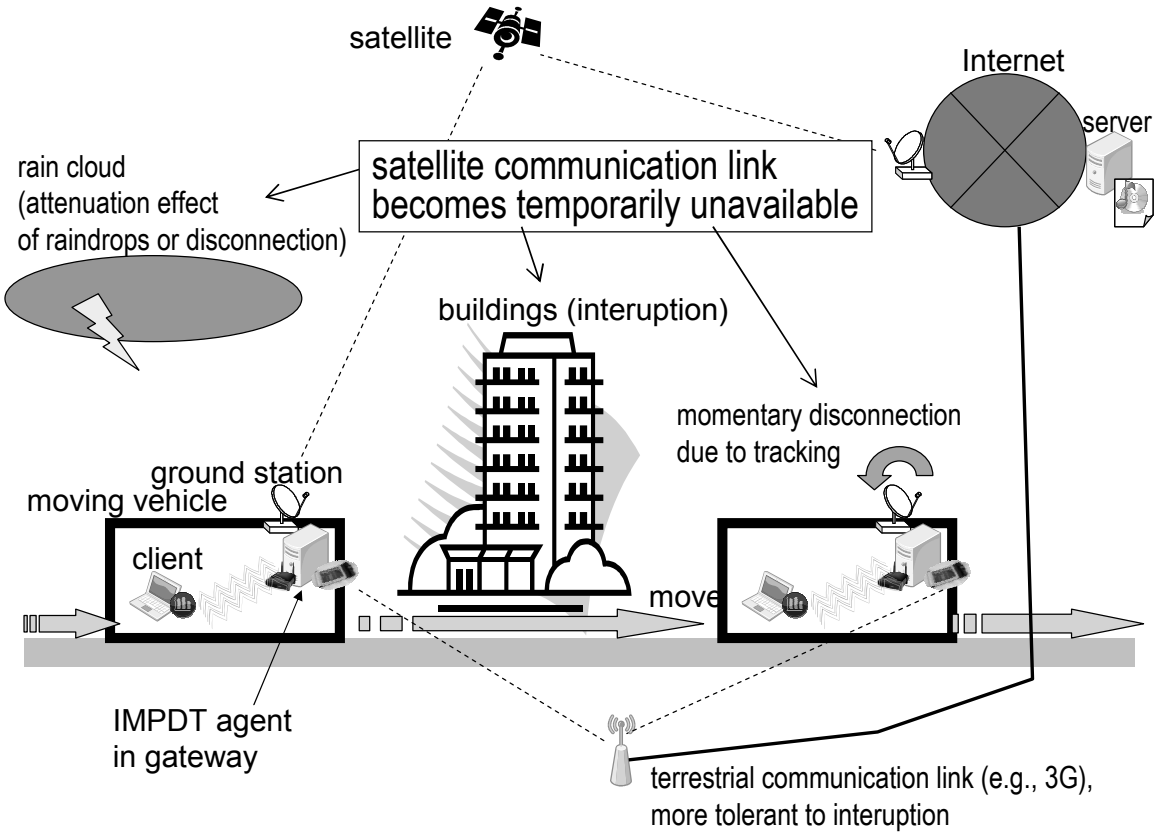


Figure 3.4: Use case example of IMPDT (2)

server during the file transfer, and is tolerant enough against environmental changes to keep information exchange while the data rate is lower. The narrowband terrestrial link can be used concurrently with the broadband satellite link to increase the sustainability.

By simplifying both examples, we can consider that two types of communication media ($NW_1$ and $NW_2$) as access networks are available to transfer a large data set (hundreds of Mbytes). $NW_1$ is a stable network with a low data rate (e.g. dozens of Kbps). Here, *stable* means continuously available for a long time and in a wide area. $NW_2$ has sufficiently high data rate but is unstable; intermittently disconnected, for example. The proposed IMPDT takes advantage of integration of such multiple, diverse, heterogeneous networks, not simply for bandwidth aggregation but also particularly for providing sustainable control informa-

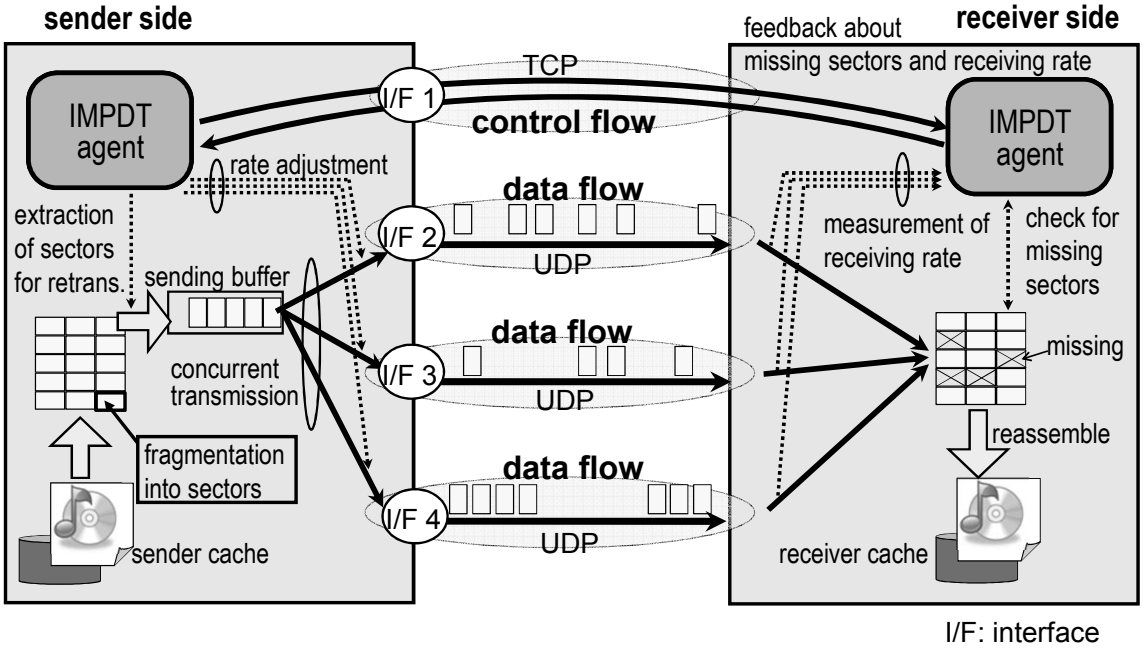


Figure 3.5: System overview of integrating multi-path data transfer (IMPDT)

tion exchange (via a stable network such as $NW_1$) for end-to-end reliable transfer. In our approach, file transfer is performed by multiple separated communication flows for control information exchange to support reliable and efficient data transmission (by control flow) and for data transmission itself (by data flow). Each flow is transmitted separately and can be conveyed by different communication media, depending upon the characteristics of the flow and communication medium. A control flow conveys information for retransmission (ACK (acknowledgement) or NACK (negative acknowledgement) ), rate adjustment, and so on, which does not require a high data rate in general.

## 3.2 IMPDT System

### 3.2.1 System Overview

To achieve efficient large-sized data transfer over challenged networks, two agents substitute for direct end-to-end communication: one is a client that initiates data transfer and the other is a server that responds to a client request. The system supports both senders and receivers in challenged networks. The client agent acts as a sender and the server acts as a receiver for file upload, and vice versa for download. It is assumed that the agent host in a challenged environment can simultaneously access multiple wireless networks of different types to take advantage of the combination and that at least one is stable even if its data rate is low. A file is transferred between these two agents by exploiting multiple available wireless networks.

The control flow is set up between agents over a TCP connection via a stable network. The file to be transmitted is then fragmented and sent out concurrently over UDP (User Datagram Protocol) flows via multiple available networks according to the conditions of the data medium. Data transmission between agent hosts uses UDP-based transmission, which attempts to send as much data as possible via each data link despite its disrupted or lossy condition. The application layer adjusts and controls UDP transmission on the basis of the feedback information exchanged via the control path. Missing sector information enables retransmission control, and received data rate information enables rate adjustment control. Note that the stable network is not dedicated for control flows alone, but can also be used concurrently for data flows.

Figure 3.5 shows the case in which the sender agent (left side) transfers a file to the receiver agent (right side), which has an interface to access a stable network and multiple interfaces (e.g., three interfaces in the figure) for accessing unstable higher-data-rate ones. The procedure of multi-network data transfer between the client and server agent in the prototype system is described as follows:

- The two agents attempt to synchronize the files stored in their local caches.

The client agent sets up a control connection to the server agent when a file transfer request occurs. The agents periodically check the synchronisation of the cached file with each other via the stable control path. If an agent finds any files in its own cache that are not found in the other agents's cache, then it reports the details of those files to the other agent.

- Files to be transferred are fragmented into sectors of a fixed size and synchronized by sector. Each sector is identified uniquely within a scope of that file, by a sequential number. The sender agent reports the number of sectors composing the file to be transferred to the agent on the receiver side.

- By the procedure described in section 3.2.2, the sender agent recognizes the list of unreceived sector identifiers. Note that in the initial phase of transfer, all sectors are regarded as unreceived until reported otherwise. Specified sectors are copied into the send buffer and transferred to the receiver agent over a set of data links, which is decided based on previous configuration and the procedure described in section 3.2.4.

- Multiple sectors can be contained in a UDP payload as long as their total size does not exceed it. The sector size is set to 256 bytes in our prototype system.

- During a file transfer, the sender and the receiver agents cooperate to control retransmission, the transmission rate of each data flow, and data-flow setup, by the procedure described in Sections 3.2.2 and 3.2.3.

- The receiver agent stores the received sectors and reassembles all the fragmented data into the original file. The file transfer finishes when the receiver agent receives all of them and reassembles the original file. After transfer between the agents, the receiver agent uploads the reassembled

file to the destination remote host, according to the upload information, by traditional means such as FTP (File Transfer Protocol).

### 3.2.2 Retransmission Control

The retransmission is based on explicit feedback, including the result of the receiver agent's check for unreceived sectors. To trigger this check, the following combination of two methods is used.

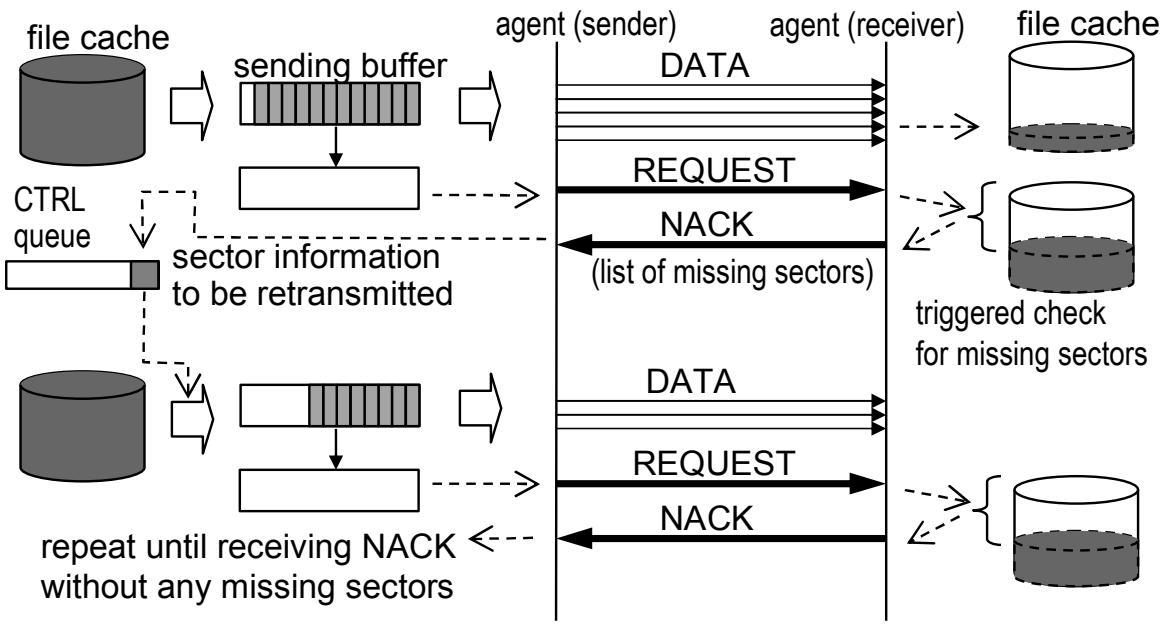


Figure 3.6: Retransmission control 1: *basic request-response style*

#### Basic Request-Response Style

A sender agent prompts the receiver agent to check whether the receiver agent has any unreceived sectors.

The sender agent generates and sends a control message (REQUEST) to the receiver agent in order to inquire the list of unreceived sectors. When the receiver agent receives

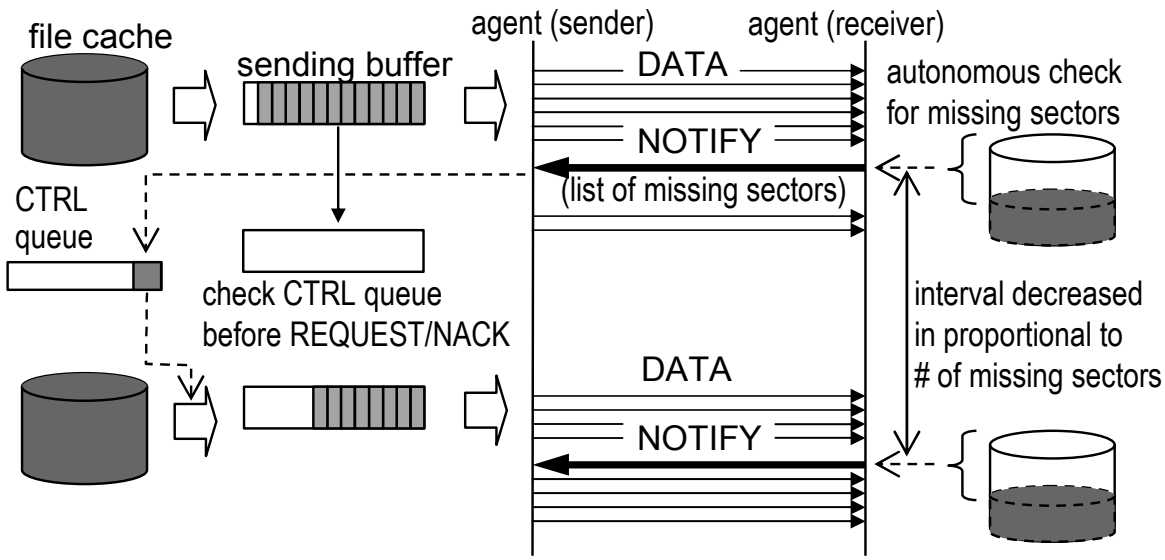


Figure 3.7: Retransmission control 2: *receiver-initiated notification*

a REQUEST message, it checks whether a specified file has any unreceived sectors. REQUEST message is generated when the send buffer is empty. The emptiness of send buffer occurs at the first step of a transfer or after all sectors in the send buffer are sent out to the receiver agent. In fact, it notifies the receiver agent that no more sectors will be sent from the sender agent. Therefore sectors unreceived at the receipt of REQUEST message are regarded as lost sectors, which must be retransmitted.

To respond to a REQUEST message, the receiver agent generates a control message (NACK), which includes a list of identifiers of the unreceived sectors. A sender agent extracts sectors specified by the list in the received NACK message, and puts them into the send buffer.

A NACK message with no sector identifier implies that all sectors are successfully received at the receiver agent, which signals the end of a transfer. The sender agent repeats this procedure until receiving such an NACK message.

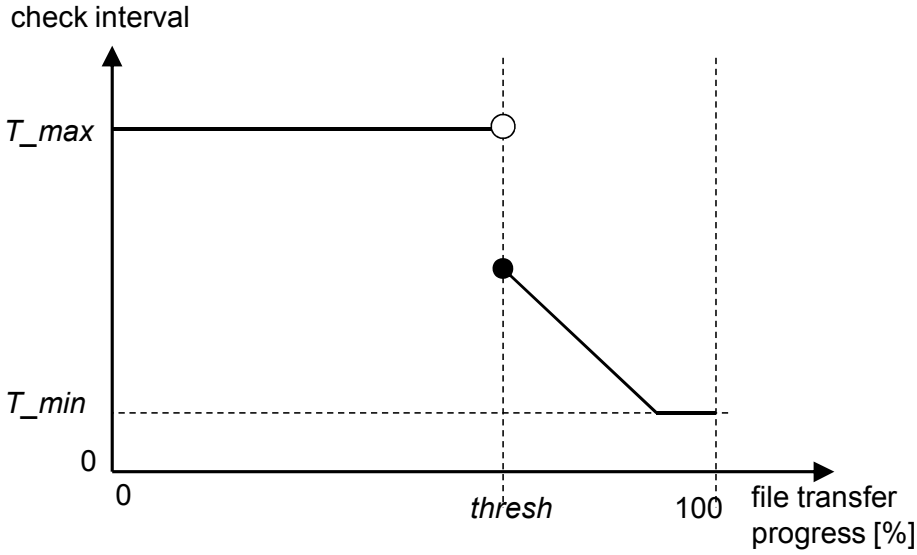


Figure 3.8: Interval of receiver-initiated notification

**Receiver-Initiated Notification**

When the number of remaining unreceived sectors becomes small, i.e. in the final stage of transfer, a sender agent may experience a long transmission idle owing to the wait for the round-trip request-response procedure if the control path has a long delay. Moreover, if the network conditions of data paths are leaky, such a retransmission with idle time may occur several times. Therefore, in the basic request–response style, a sender agent could suffer from this last–segments transmission idle.

To improve the performance degradation of this transmission idle, a receiver-initiated notification method is also introduced. As illustrated in Figure 3.7, this method is a self-directed check of unreceived sectors by the receiver agent, independent of the reception of REQUEST messages generated by the sender agent. The receiver agent autonomously checks for unreceived sectors of the received file during the transfer. The check is performed at certain intervals. The receiver agent generates a control message (NOTIFY) to notify the sender agent of the results of the autonomous check. A NOTIFY message includes the

same content as a NACK message. A sender agent regards sectors listed in this message as lost sectors and copies them into the send buffer for retransmission. Note that the sender agent reacts to the received NOTIFY message only when its send buffer becomes empty. If multiple different NOTIFY messages are received before the send buffer becomes empty, the sender agent processes only the most recently received message and discards the out-of-date ones.

In this method, the autonomous check interval can be equivalent to a retransmission timeout. A shorter interval tends to trigger aggressive retransmission, but if it is too short, it often causes excess retransmission during the initial stage of transfer. If it is too long, it may contribute little to the improvements to the degraded performance. The autonomous check interval can be changed adaptively so as to decrease in proportion to the percentage of unreceived sectors. The final stage starts when the percentage of successfully received sectors (i.e., file transfer progress) exceeds a certain threshold *thresh*. As shown in Figure 3.8, the interval of the autonomous check at a receiver side $I$ is calculated as follows:

$$
\begin{aligned}
&if(P < thresh)\\
&\quad I = T_{max}\\
&else\\
&\quad I = max[\{(1-P)\times a\}, T_{min}]
\end{aligned}
$$

Here, $P$ denotes the file transfer progress. $T_{max}$, $T_{min}$, and $a$ are parameters. We currently use static configured values for these parameters including *thresh*. In our future work, we intend to determine them dynamically and automatically according to the status of the transmission and networks.

### 3.2.3 Transmission-Rate Control

In challenged environments, it is necessary to fully and efficiently use valuable network resources that are not always available because of frequent disconnection or severe changes in bandwidth. It is efficient that data packets are sent out with as high data rate as possible in order to achieve better transfer performance. At the same time, however, it is necessary to avoid an overloaded transmission rate which would cause excess retransmission and would lengthen the transfer time.

The proposed system controls the transmission rate of each of multiple data flows independently. A sender agent cooperates with a receiver to estimate the bottleneck available bandwidth of the end-to-end path through which each data flow travels. Figure 3.9 shows that a sender agent controls its sending rate on data flow 1, which is one of data flows used in multi-path transfer. A receiver measures the averaged receiving rate of each data flow and gives feedback about its rate information to a sender agent by a control message (STATUS) via control path. Basically, the sender agent adjusts the data flow's transmission rate according to the measured receiving rate seen in the STATUS message. As shown in Figure 3.9 (a), if a receiver detects that the receiving rate $Y$ is lower than $X$ measured before, a receiver sends a STATUS message indicating the newly measured rate $Y$. A sender changes its sending rate from $X$ to $Y$.

The available bandwidth may change dynamically. To follow its change and transmit data flow at as high a data rate as possible, the sender agent additionally makes periodical bursty transmission with a higher data rate than that adjusted by the STATUS message. If the available bandwidth of the network path increases after a previous adjustment, this bursty transmission makes the receiver agent measure and give feedback of a higher rate than before. As shown in Figure 3.9 (b), when a receiver detects that the averaged receiving rate $Z$, led by burst transmission of a sender, is much higher than $Y$ measured before, the receiver sends a STATUS message indicating the measured rate $Z$.

In each case, a STATUS message is generated only when receiving rate of a data flow

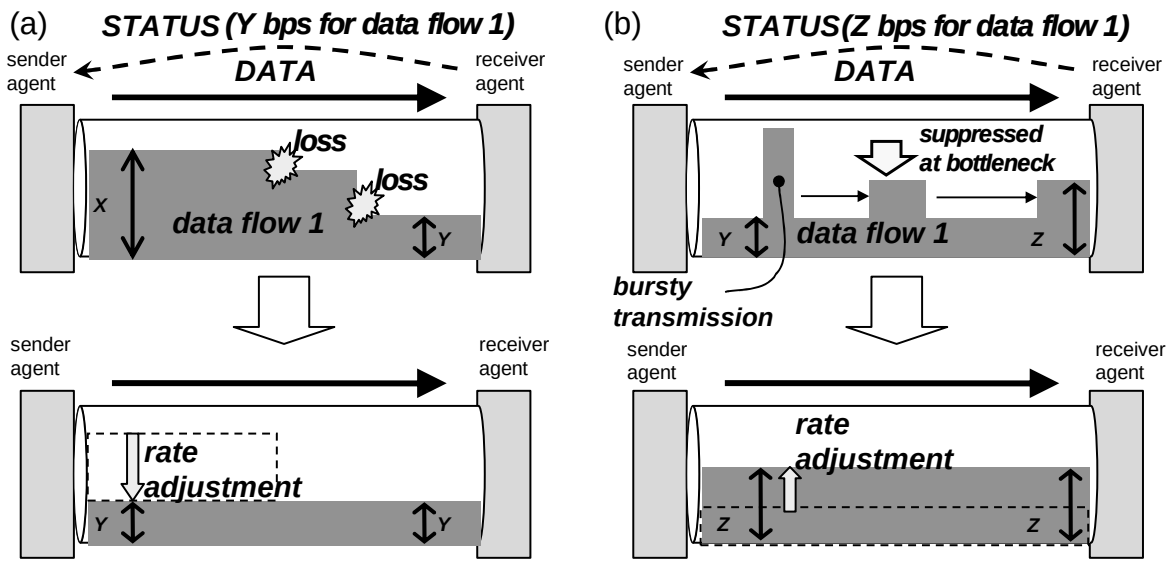


Figure 3.9: Transmission rate control

changes at a certain level to avoid oscillation by too frequent changes. Note that this approach to estimating the end-to-end available bandwidth is not always accurate. However, since we use an averaged receiving rate of a sequence of the burst test packets and also check losses of those packets, our method can be usable for the purpose of triggering an increase of the sending rate. To implement a more accurate estimation method is one of our future works.

### 3.2.4 Data-Flow Setup Control

When a receiver agent has multiple interfaces desired to receive data flow, a sender agent tries to transfer with multiple data flows to exploit multiple network accesses to a receiver. An issue arises that it is difficult for a sender agent to understand the status of data interfaces at a receiver agent: IP address or connectivity (up or down). It is especially true, for an example, that a client agent in the widely moving vehicle requests download and plays a receiver role. In this case, an IP address of data interface of a client agent may dynamically and often change.

As shown in Figure 3.10, in our proposed system, a receiver agent generates a control message (TOPOLOGY) including list of a set of device name (e.g., ppp0 or wlan0), IP address and connectivity information of a data interface. This message is generated not only at the start of transfer, but also whenever an IP address or its connectivity is changed. TOPOLOGY message is sent to a sender agent. Note it is assumed in this paper that which device to be used for receiving data flow has been previously configured.

The TOPOLOGY message enables a sender agent to immediately follow the change of interface status of a receiver agent. A sender agent reacts according to the contents of the received TOPOLOGY message. When the connectivity status of a device that has never been up becomes up, it sets up a new data flow targeted to the specified destination IP address. When a device that has been up becomes down, it removes the corresponding data flow. When an IP address of a device changes in TOPOLOGY message, it takes away corresponding data flow and sets up a new data flow targeted to the specified destination IP address.

## 3.3 Basic Evalutation

Table 3.1: Characteristics of emulated network

| | path | bandwidth | delay | loss probability |
|---|---|---|---|---|
| $NW_0$ | control | 64Kbps | 150ms | 0% |
| $NW_1$ | data 1 | 11Mbps | 25ms | random 5% |
| $NW_2$ | data 2 | 3Mbps | 25ms | random 10% |

We present a basic evaluation of the prototype system to transfer a 100M-byte file between two agents in an indoor experiment. As shown in Figure 3.11, three different emulated networks ($NW_0$, $NW_1$, and $NW_2$) are available for a sender agent. The characteristics of each network are shown in Table 3.1. While the characteristics of $NW_0$ are low speed and long

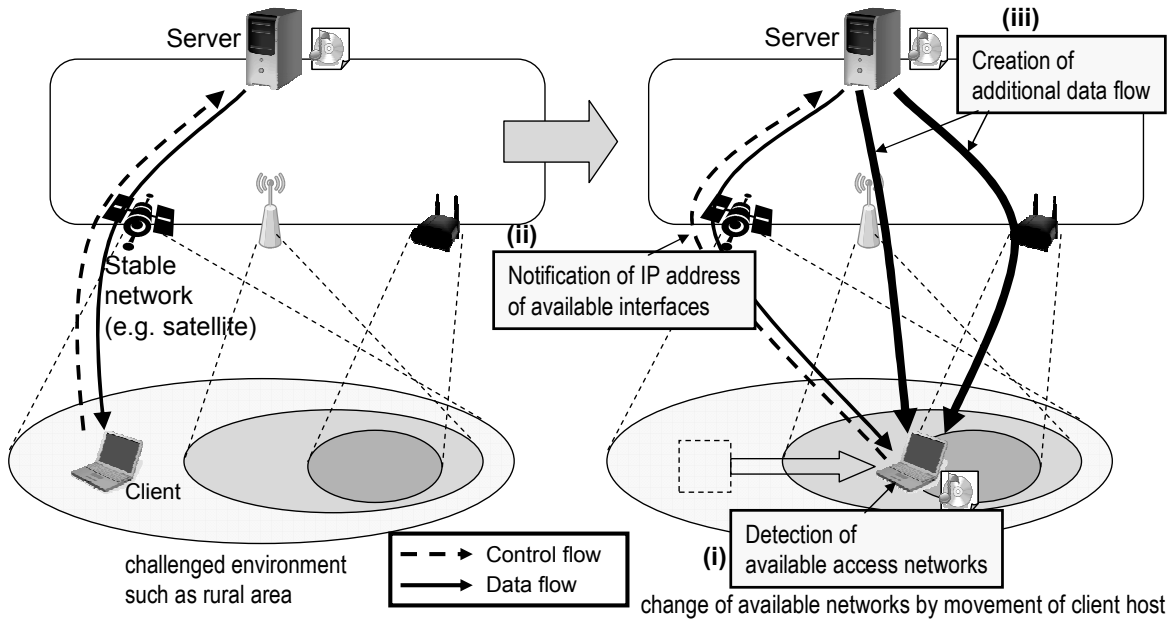


Figure 3.10: Topology control

delay, but very stable (no loss), those of $NW_1$ and $NW_2$ are higher speed but very leaky. Therefore, $NW_0$ is used for the control path, and the others are used for the data paths.

### 3.3.1 Aggregation of Data Paths

Figure 3.12 shows the result of a 100M-byte file transfer using a separated control flow ($NW_0$) and two data flows ($NW_1$ and $NW_2$). The left y-axis shows the receiving rate of the data flow. The right y-axis shows the percentage of successfully received sectors as the file transfer progresses. The file-transfer progress reaches 100% at the completion of the transfer. Both of the methods shown in Sections 3.2.2 and 3.2.2 are used for retransmission. The threshold for enabling the receiver-initiated notification (autonomous notification of a receiver agent) is set to 70. The average throughput of data flows 1 and 2 is 8.76 Mbps and 1.98 Mbps respectively. On the other hand, Figures 3.13 and 3.14 show the results of the cases using only one data flow ($NW_1$ and $NW_2$, respectively) and a separated control flow.

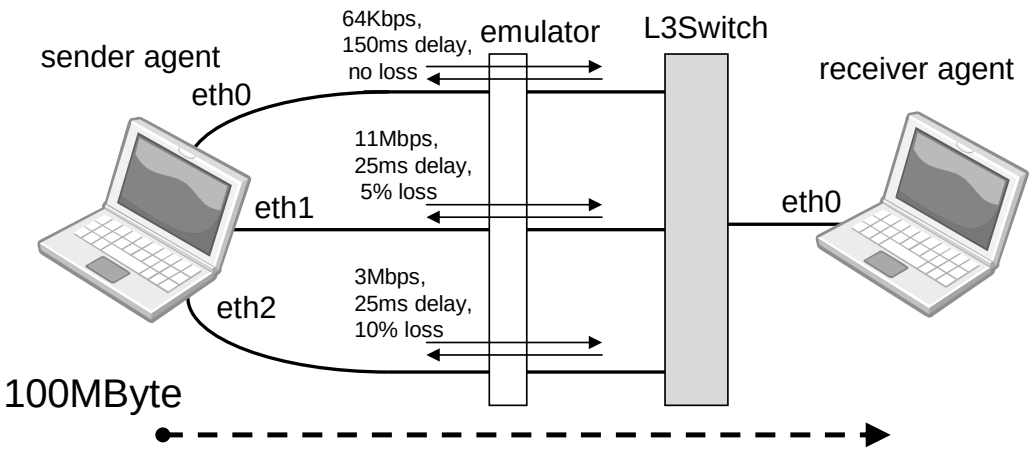


Figure 3.11: Configuration of emulated network

The average throughput of data flow 1 in Figure 3.13 is 9.08 Mbps. That of data flow 2 in Figure 3.14 is 2.12 Mbps. This experiment demonstrates that our proposed system effectively utilizes the bandwidth of multiple data paths by aggregation.

Table 3.2: An average transfer time using two data paths

| Control-path | (1) no loss 150ms delay | (2) no loss 500ms delay | (3) 10% loss 150ms delay |
|---|---|---|---|
| Retrans. w/o NOTIFY | 81.0 s ($\sigma$ =6.2) | 97.4 s ($\sigma$ =9.3) | – – |
| Restans. w/ NOTIFY | 79.6 s ($\sigma$ =2.7) | 80.4 s ($\sigma$ =2.5) | 123.2 s ($\sigma$=31.3) |

### 3.3.2 Impact of Control Path Characteristic

This section shows the evaluation on how the difference of control path characteristics impacts on the transfer performance. The transfer time of a 100M-byte file is evaluated in the

same configuration as that of Section 3.3.1, except for the delay and loss probability of $NW_0$.

First, we focus on the impact of control delay only and assume that no loss occurs in control path. As stated in Section 3.2.2, a longer control delay will lead to a large transmission idle on the sender side without a receiver-initiated notification. Table 3.2 (1) and (2) show a comparison of the transfer time with or without a receiver-initiated notification in a different control-path delay, 150 ms and 500 ms. $\sigma$ shows the standard deviation. It shows approximately the same average transfer time in both the cases with NOTIFY (79.6 s) and without NOTIFY message (81 s), when the control-path delay is 150 ms. However, when the control-path delay is 500 ms, it takes 80.4 s in the case with NOTIFY but 97.4 s in the case without NOTIFY. This difference might come from the transmission idle time of the sender at the final stage of transfer. The introduction of NOTIFY suppresses its performance degradation.

Second, we focus on the impact of control-path stability, i.e. loss probability in control-path. Table 3.2 (3) shows the transfer time when the loss probability of the control-path is equal to 10%. The average transfer time is 123.2 s, and the standard deviation is 31.3. As compared to (1), they are much larger and dispersed. When a control-path suffers from heavy packet losses, some STATUS or NOTIFY message are lost or significantly delayed. This causes a sender to fail to perform an appropriate rate control or retransmission, which results in a longer and more variable transmission time.

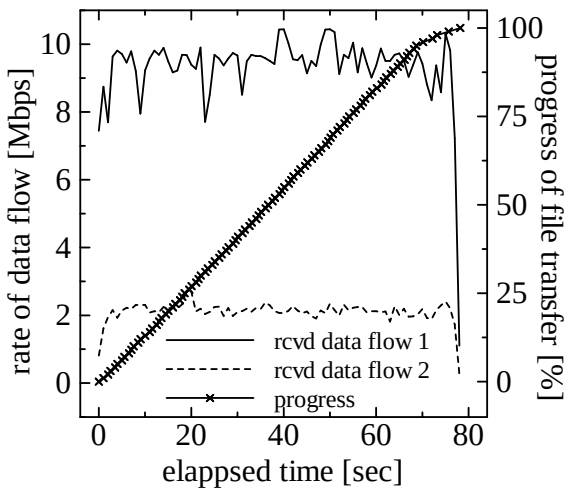


Figure 3.12: Result of emulated network using two data flows (data path 1 and 2)

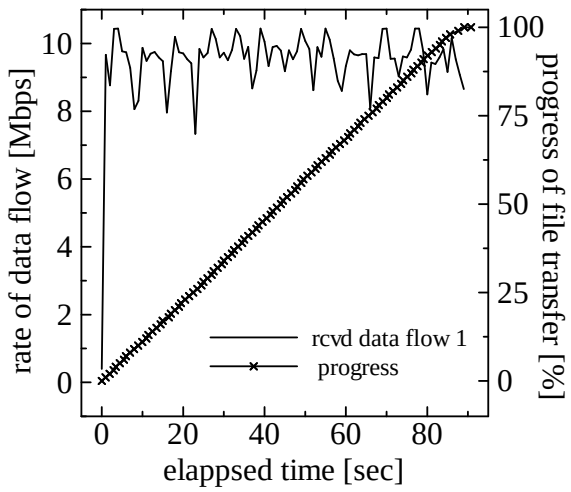


Figure 3.13: Result of emulated network using one data flow (data path 1)

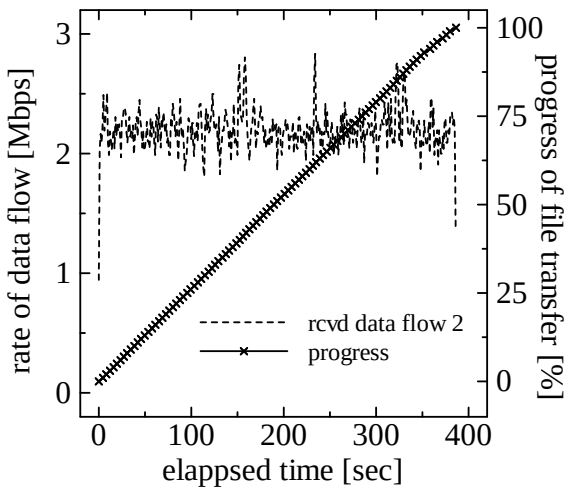


Figure 3.14: Result of emulated network using one data flow (data path 2)

# Chapter 4

# IMPDT Experiments using Real-World Communication Network

## 4.1 Overview of Experimental Examinations

This chapter describes three field experiments using a combination of terrestrial communication links and a satellite communication link, in order to validate our prototype implementation including basic request-response, receiver-initiated retransmission control, and transmission-rate control.

The first experiment was conducted in May 2008, and used a satellite communication link, ETS-VIII[32], which was regarded as a *low-speed but stable* network and, because of its mobility awareness, was better suited for control flow in that experiment.

The second experiment was conducted in June 2010, and used a high-speed satellite communication link, WINDS (Wideband InterNetworking engineering test and Demonstration Satellite)[35], which was regarded as a *higher-speed but unstable* network that was better suited for data flow.

The third experiment was conducted in November 2009, and used a commercial global satellite communication network which was attached as in-vehicle base-station antenna. The

car moving around with the client agent inside to examine a moving vehicle situation. This satellite communication service covers all aspects of this experiment with low data rate, which was regarded as a *low-speed but stable* network.

## 4.2 Experiments Using Low-Speed but Stable Satellite Communication Link

### 4.2.1 Experimental Configuration

We tested the prototype system in a file transfer experiment between two locations in Kitakyushu (Fukuoka) in Japan. As shown in Figure 4.1, the control channel is setup through a satellite link of the engineering testing satellite ETS-VIII. The data links are an IEEE802.11g Wi-Fi and a cellular Internet service. The file upload source moves over several Wireless Access Points (APs) during data upload, which results in disconnection and disruption within the Wi-Fi link . In this experiment, agent node at site A switches its associating AP between AP1 and AP2 every 30 seconds. While staying at an AP area, the transmission rate of its wireless link changes from 5.5 Mbps to 54 Mbps, and instantaneous interruption also occurs. These conditions of the Wi-Fi link are manually operated to simulate the scenario as shown in Figure 4.2. In this scenario, we simulated a mobile host moving across multiple Wi-Fi hotspot areas. The cellular link (384 Kbps uplink) has high latency, and its delay is not constant. In this section our main goal is to validate if our prototype works well on an agent host installed multiple interfaces each of which has different type of characteristic in the real environment. This system was proven to perform as expected in the actual environment.

### 4.2.2 Experimental Results

Experimental combinations of media and functions (data/control) shown in Table 4.1 are examined for performance comparison. Cases 1a and 1b correspond to the traditional case

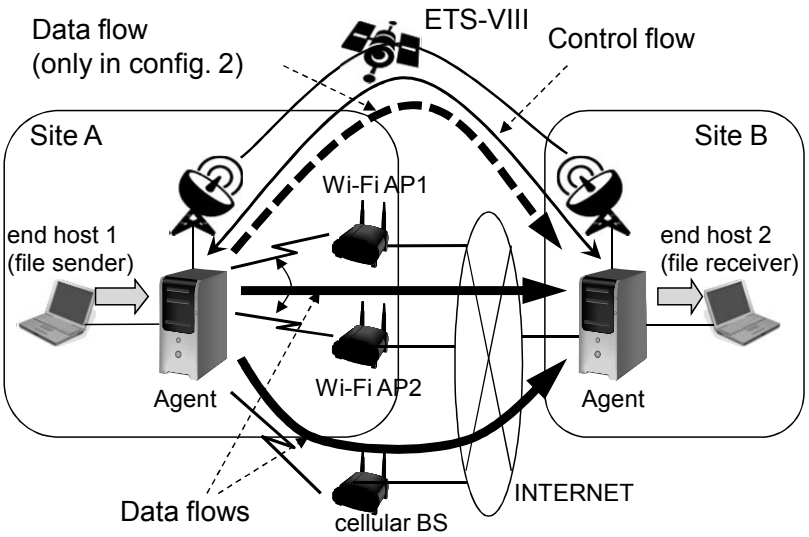


Figure 4.1: Configuration of experiment using ETS-VIII

in which an application uses two (in configuration 1) or three (in configuration 2) parallel TCP connections at the same time to fully utilize the available media (communication paths). Case 1a evaluates CUBIC TCP[33] which is widely used in current Linux kernels by default. Case 1b evaluates TCP westwood+[34] which is well known as an enhanced version for wireless communication. Case 2 represents the proposed IMPDT system with two (in configuration 1) or three (in configuration 2) UDP data flows and control channel via satellite link. Figure 4.3 shows an example of the experimental results in configuration 1. It evaluates the time required to transfer a 100-Mbyte file from an agent node at site *A* to one at site *B*. The result of Case 1a shows the summation of the amount of data transferred by each of two independent CUBIC TCP connections. It takes 237 seconds to transfer 100-MByte data. It takes 227 seconds even by two independent TCP westwood+ connections (Case 1b). Contrastively, in the proposed IMPDT shown as Case 2, it takes 116 seconds only, around half of Case 1. Similarly, Figure 4.4 shows an example of the experimental results in configuration 2. It takes 217 seconds to transfer 100-MByte data by three independent CUBIC

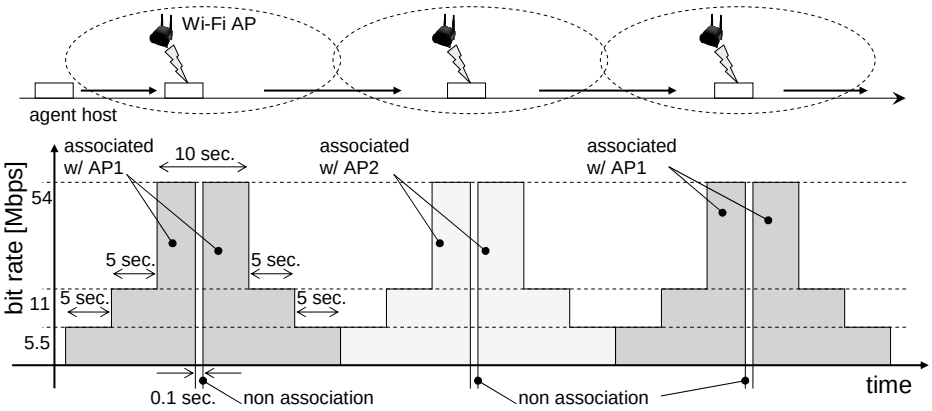


Figure 4.2: Wi-Fi scenario in ETS-VIII experiment

Table 4.1: Test cases of experiment using ETS-VIII

| Case | Flows (config. 1) | | | Flows (config. 2) | | |
|---|---|---|---|---|---|---|
| | via ETS-VIII | via Wi-Fi | via 3G | via ETS-VIII | via Wi-Fi | via 3G |
| 1a | – | Cubic TCP | Cubic TCP | Cubic TCP | Cubic TCP | Cubic TCP |
| 1b | – | TCP westwood+ | TCP westwood+ | TCP westwood+ | TCP westwood+ | TCP westwood+ |
| 2 | IMPDT (CTRL) | IMPDT (DATA) | IMPDT (DATA) | IMPDT (CTRL+DATA) | IMPDT (DATA) | IMPDT (DATA) |

TCP connections (Case 1a). It takes 199 seconds even by three independent TCP westwood+ connections (Case 1b). Contrastively, in the proposed IMPDT shown as Case 2, it takes 107 seconds only, around half of Case 1. Especially on a Wi-Fi link, while the rate of Case 2 increases according to the change in the bit rate of Wi-Fi when a higher bit rate becomes available, the rate of Case 1 increases much less than the rate of Case 2. We consider that this is possibly because the TCP connection suffers from instantaneous interruption during a high bit rate and/or in frequent AP switching, and can not increase its sending rate adequately.

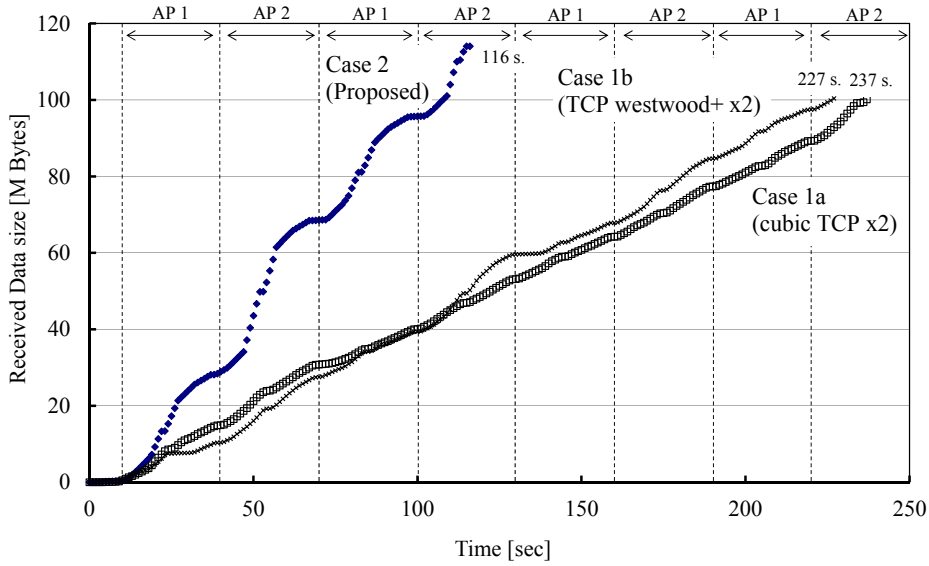


Figure 4.3: Results for configuration 1 of experiment using ETS-VIII

## 4.3 Experiments Using High-Speed but Unstable Satellite Communication Link

### 4.3.1 Experimental Configuration

To test our prototype system in different scenarios, we conducted another experiment using WINDS (Wideband InterNetworking engineering test and Demonstration Satellite)[35], which is regarded as a *higher-speed but unstable* network in contrast to the satellite link in the previous subsection, to transfer large files between two locations in Japan in June 2010. WINDS aims at providing engineering test and demonstration environments for large-capacity data communications by satellites in such fields as Internet communications, education, medicine, disaster measures, and so on.

As shown in Figure 4.5, a client agent located in Kashima had two available networks to a server located in Koganei: a satellite link and a terrestrial cellular link. We tested the four cases shown in Table 4.2. In Cases 1 and 2, the client in Kashima downloaded a 100-Mbyte

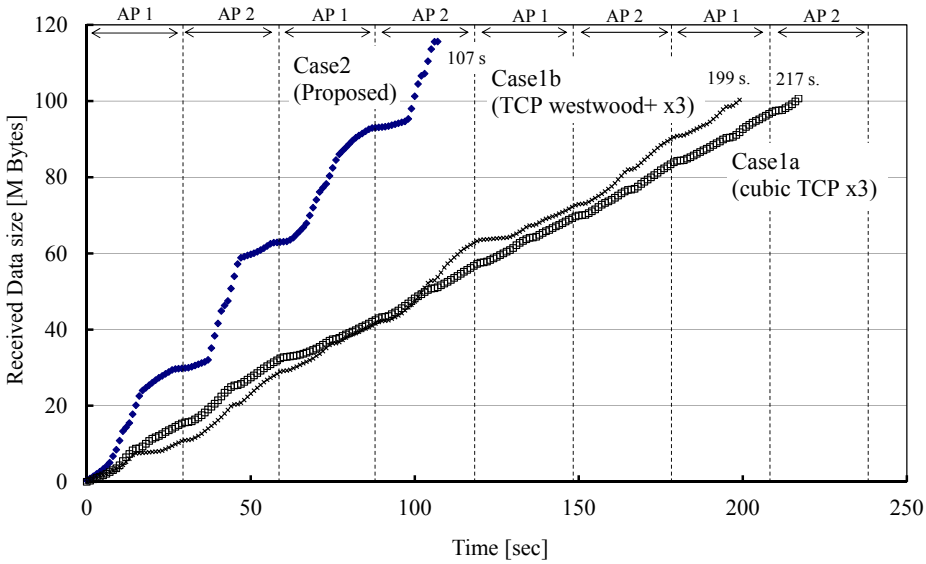


Figure 4.4: Results for configuration 2 of experiment using ETS-VIII

file from a server in Koganei, i.e., the direction of file transfer was from Koganei to Kashima. In Cases 3 and 4, a client uploaded the file to a server.

In Cases 1 and 2, the transfer times of the file were almost the same. These results indicate that our prototype works well in both cases and there is no negative impact, even if the control flow travels through a different network.

### 4.3.2 Experimental Results

Figures 4.6 and 4.7 shows the results for Cases 3 and 4, respectively. The left y-axis shows the sending (dotted-line) and receiving (solid-line) rates of the data flow. The right y-axis shows the percentage of successfully received sectors as the file transfer progressed. Incidentally, the satellite link condition was lossy in Cases 3 and 4. This may be primarily due to bad weather around Kashima when Cases 3 and 4 were tested, which may have caused unexpectedly strong rain-induced signal attenuation. The result for Case 3 indicates that the data flow sending rate was as high as possible, despite the lossy condition of the satellite

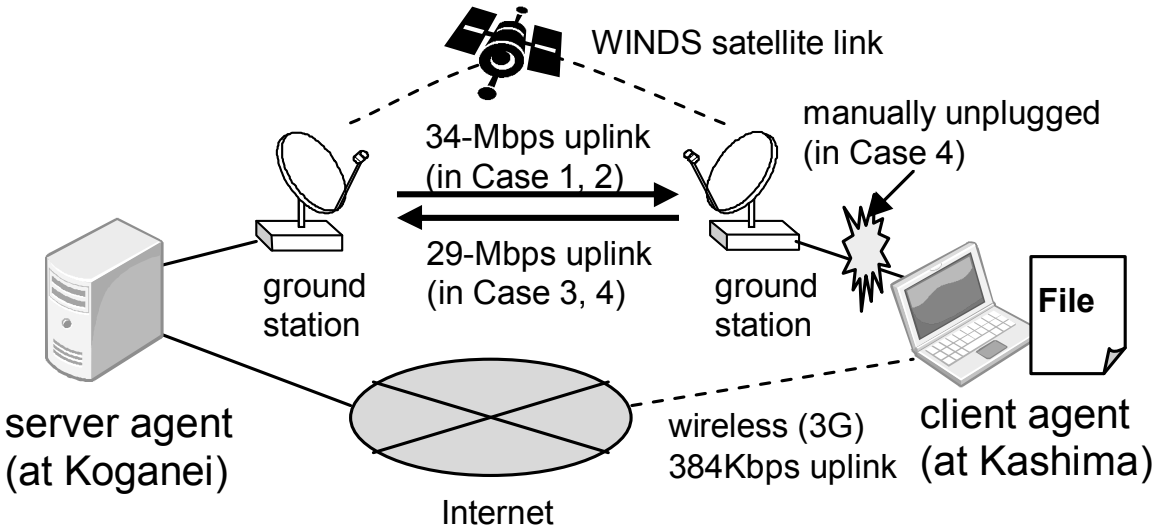


Figure 4.5: Configuration of experiment using *high-speed* satellite link (WINDS)

link. This is due to the steady exchange of control information via the separate 3G link. In Case 4, in order to allow the file transfer in a challenged condition, the intentional disruption was applied by manually unplugging the Ethernet cable between the ground station and the client agent approximately 15 seconds after the start of transfer. As plugged it in again after a short interval, the IP connectivity recovered in approximately 55 seconds. The data flow through the satellite link stopped and restarted according to its connectivity status. These results indicate that as the stable 3G link maintains the control flow despite the satellite link disconnection. The client (sender) agent can be notified of the disconnection and recovery of satellite link connectivity by feedback from the server (receiver) agent via the control flow.

## 4.4 Experiments in a Moving Vehicle Environment

### 4.4.1 Experimental Configuration

Table 4.2: Test cases of experiment using WINDS

| | Direction of Transfer | Flows | |
|---|---|---|---|
| | | via Satellite | via 3G |
| Case 1 | Koganei → Kashima | CTRL, DATA | – |
| Case 2 | (download scenario) | DATA | CTRL |
| Case 3 | Koganei ← Kashima | DATA | CTRL |
| Case 4 | (upload scenario) | DATA (w/ disruption) | CTRL, DATA |

We tested our prototype system in a field experimental environment in Miyazaki, Japan, in November 2009. The environment was originally organized for field experiments of construction of an ad hoc temporary network access service in disaster fields (especially in a rural area) where no network infrastructure is available. A temporary balloon wireless mesh network [36] was constructed in that field. Balloon nodes with Wi-Fi access point devices were launched to organize a multi-hop mesh network in the sky and to provide a hot-spot Wi-Fi service to the ground area below. This multi-hop balloon Wi-Fi network also provided a connectivity to a base building (headquarter). We used this balloon network concurrently with a commercial global satellite communication network which covers all aspects of this experiment with low data rate.

As shown in Figures 4.8 and 4.9, a client agent located in a car that moves on the road. It had two network access interfaces: Wi-Fi and satellite communication (the car had an in-vehicle base-station antenna). A server agent located in the building as a base that had stable connectivity from both the balloon wireless network and the satellite communication network through the Internet. The client agent in the car moving on the road uploaded a file to the server agent. Agents set up both control and data flow through the satellite communication network. The car with the client agent moved across the area beneath the

balloon Wi-Fi several times (three times in this trial) during file transfer. When the client agent detected Wi-Fi connectivity, it also set up a data flow through Wi-Fi to the server. The data rate of Wi-Fi, if connected, was much higher than that of satellite communication.

### 4.4.2 Experimental Results

An example of the experimental results is shown in Figure 4.10. The left and right axes show the data flow receiving rate and the percentage of successfully received sectors as the file transfer progressed, respectively. The file transfer progress reached 100% at transfer completion. When the car moved across the Wi-Fi area, the file transfer progress increased faster than during the time without Wi-Fi connection. This result shows that the two access networks were efficiently integrated. In the third Wi-Fi session, however, the file transfer progress increased less quickly than that in the previous two. This was caused by inefficient duplicated retransmission in the final phase of transfer owing to the long delay of the control path through satellite communication. More improvement is needed and this is for future work.

In some cases, we experienced that the connectivity and the data transmission throughput of the Wi-Fi network severely degraded than expected because the position of each balloon node got unstable due to the strong wind. In addition, this kind of Wi-Fi networks might be easily influenced by the weather condition. Although we could not provide quantitative measurement, we believe that the integrated use of heterogeneous networks by IMPDT is a key to achieving tolerance to such unstable conditions.

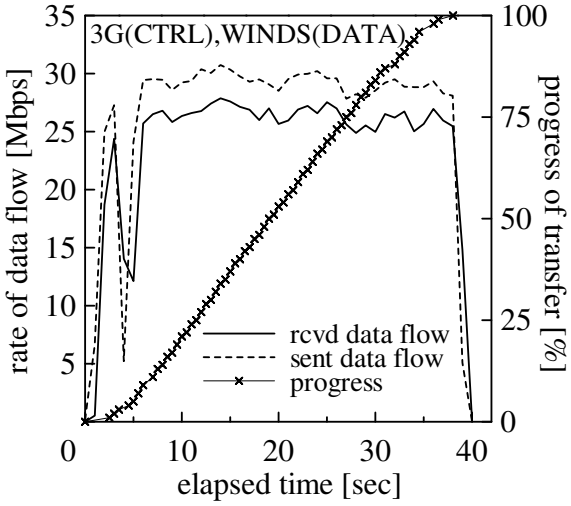


Figure 4.6: Results for WINDS experiment Case 3

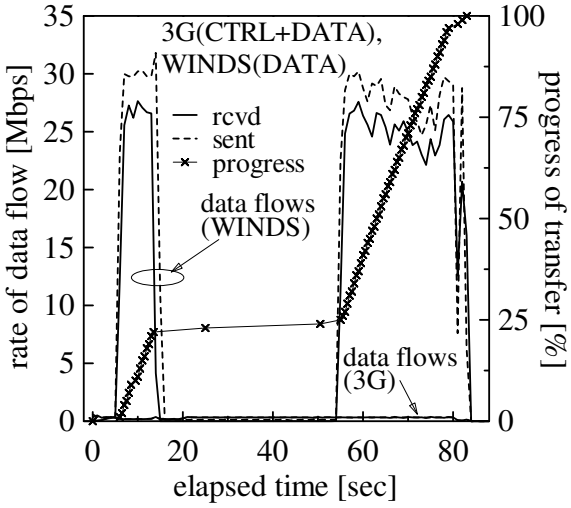


Figure 4.7: Results for WINDS experiment Case 4

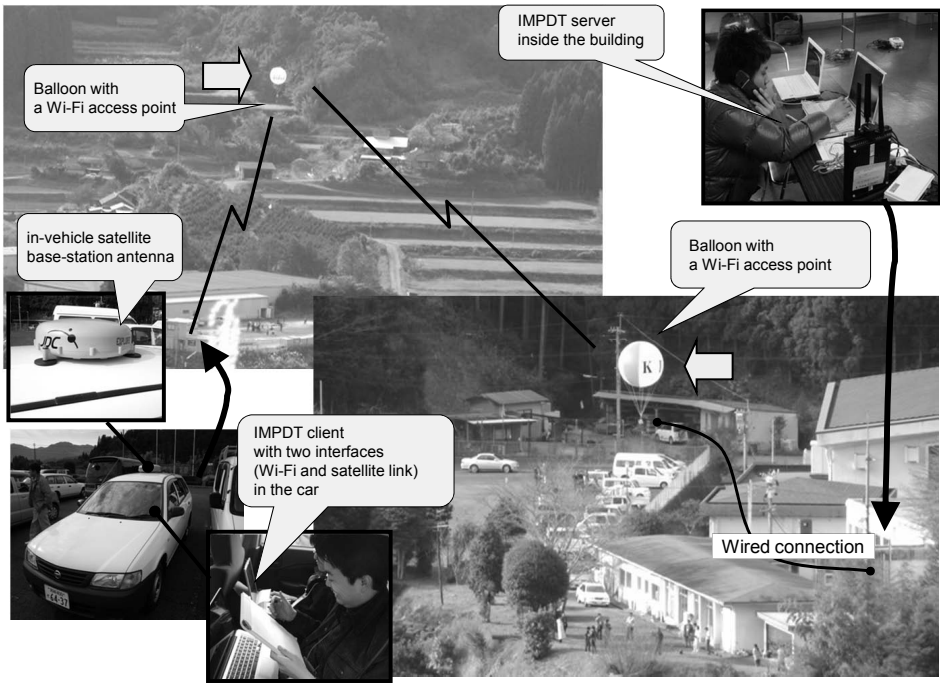


Figure 4.8: Photograph of field experiment using a *low-speed* satellite link (November 2009, Miyazaki, Japan)

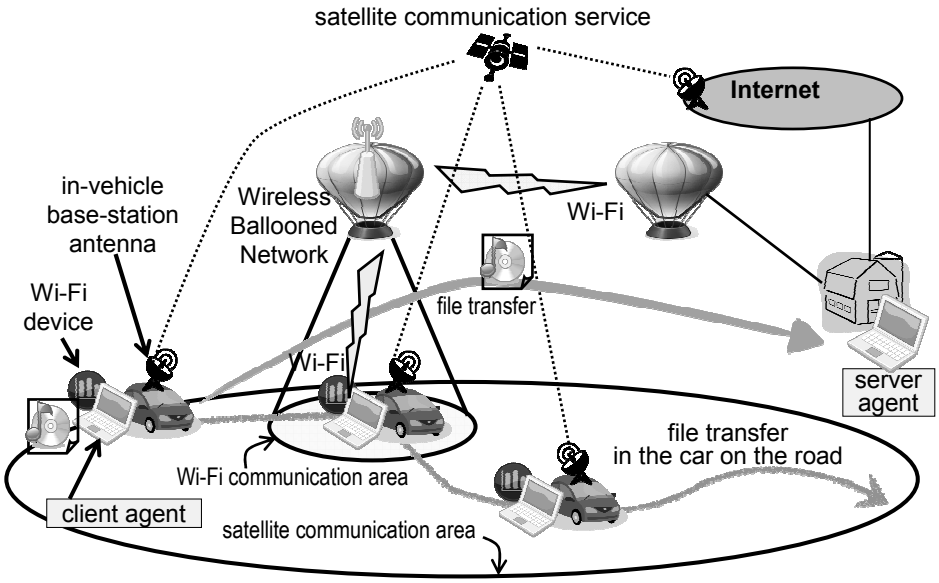


Figure 4.9: Overview of field experiment using a *low-speed* satellite link

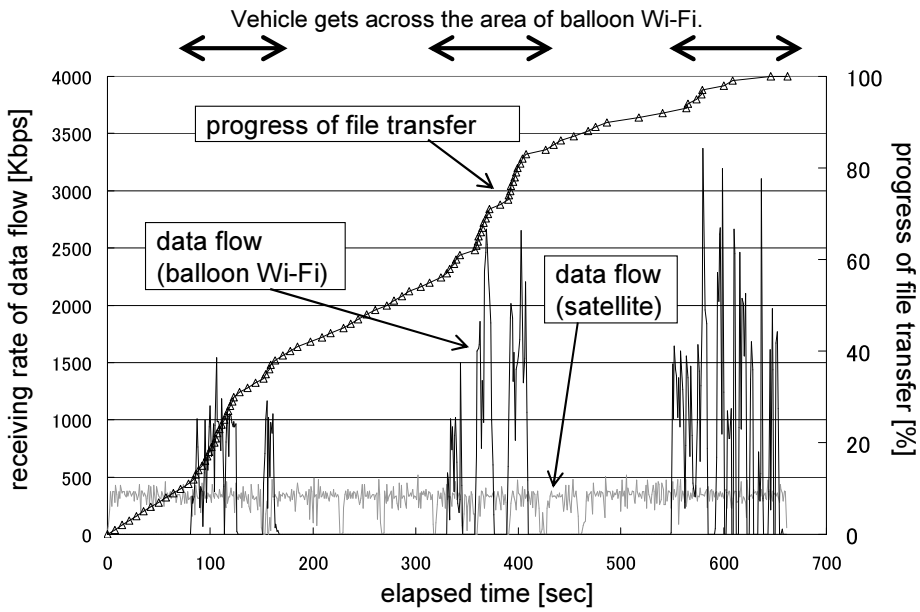


Figure 4.10: Example of experimental results

# Chapter 5

# IMPDT Coordinated with Application

## 5.1 Consideration of Integration with Interactive Application

In this chapter, we apply IMPDT to an interactive application. It is not easy to use an interactive application in challenged networking environments, because continuous connectivity during use is assumed. Drive-thru Internet [38, 39] was proposed to make web browsing possible over an intermittent network provided by underlying store-carry-forward based message transfer (DTN bundle) [37], in which a new protocol was introduced to protect an interactive application 's session against frequent network disruption. On the other hand, assuming multiple available networks—and that one of them is at least stable even although its data rate may be quite low—our scheme can achieve both a continuous connectivity to keep an application session active and efficient data download using intermittently connected but high data rate networks at the same time.

## 5.2 Web Browsing Access Using IMPDT

To expand the IMPDT scheme from non-interactive bulk transfer to interactive applications, we have designed and implemented a prototype web access system for web browsing over multiple challenged networks by integrating the HTTP protocol and the IMPDT scheme. The proposed system handles each file, consisting of a web page, differently and selects an appropriate transport-layer method for the file. The main text-based files are sent on a low data rate but stable TCP path, while the subsidiary but large contents are separately sent over multiple networks provided by the IMPDT scheme. Each web page generally consists of a number of files. One of them, the manuscript file (typically an html file), prescribes the page layout and includes text and links to other contents to be shown on the web page. The other files, including figures, images (e.g., jpg or png) and movies (e.g., mpg or mov), are contents referred to by the manuscript file. Although the manuscript file is relatively small, it is fundamental for web access because the contents files can be accessed only after the html file has been read and processed. From this point of view, the html file can be regarded as a kind of control message in the context of a web access application. In a challenged environment, such a manuscript file should be transferred separately via a stable control path of IMPDT. Other contents such as images or movies of relatively large sizes should be transferred by the IMPDT scheme.

In Figure 5.1, a user at host $A$ tries to access a web page served by a web server in the Internet through a challenged environment, where host A can access three types of communication media ($NW_0$, $NW_1$, and $NW_2$). Host $A$ has an IMPDT agent and $NW_0$ is suitable for its control path. Host $A$ also has a proxy function for web access. This proxy function works differently depending on what kind of file is being accessed.

For fetching a fundamental file (i.e., html), this proxy sends out an http request message via the control path of IMPDT (i.e., $NW_0$) in order to download that html file. For fetching contents files referred to by that html file, this proxy internally requests an IMPDT agent to obtain those files by using the IMPDT method (i.e., using $NW_0$, $NW_1$, and $NW_2$). Then, the

IMPDT agent initiates file transfer to download files requested by the proxy.

Figure 5.2 and 5.3 show an example of web access procedure. A user inputs a uniform resource locator (URL, which is assumed to be that of server $C$) into a web browser and an http request is consequently sent to $C$. The intermediate proxy analyses this request message, recognizes that it is requesting the html file, and relays it via a control path to $C$. After downloading the html file using the normal http procedure via the control path, the web browser parses it and extracts references to additional contents from it. They are also assumed to be stored on server $C$. Http request messages are again sent to download these contents. The intermediate proxy analyses them, recognizes that they requesting not a html file but images or movie contents, and requests the IMPDT agent to download them instead of relaying those http request messages. The IMPDT agent at the client side initiates the IMPDT procedure to synchronize these requested files with its peer agent at server side, i.e. host $B$. Server agent $B$ downloads the specified files from server $C$ prior to IMPDT transfer if they are not already stored in its local cache. After completion of the contents download by IMPDT transfer, the intermediate proxy sends those contents to the web browser. This completes the sequence of web access. As shown in Figure 5.4, if this proxy additionally functions as a TCP proxy such as performance enhancing proxies [40], it will also work smoothly with an internal or external web browser. In this case, the intermediate proxy sends back local ACKs to the web browser host to keep the underlying TCP connections alive during IMPDT transfer.

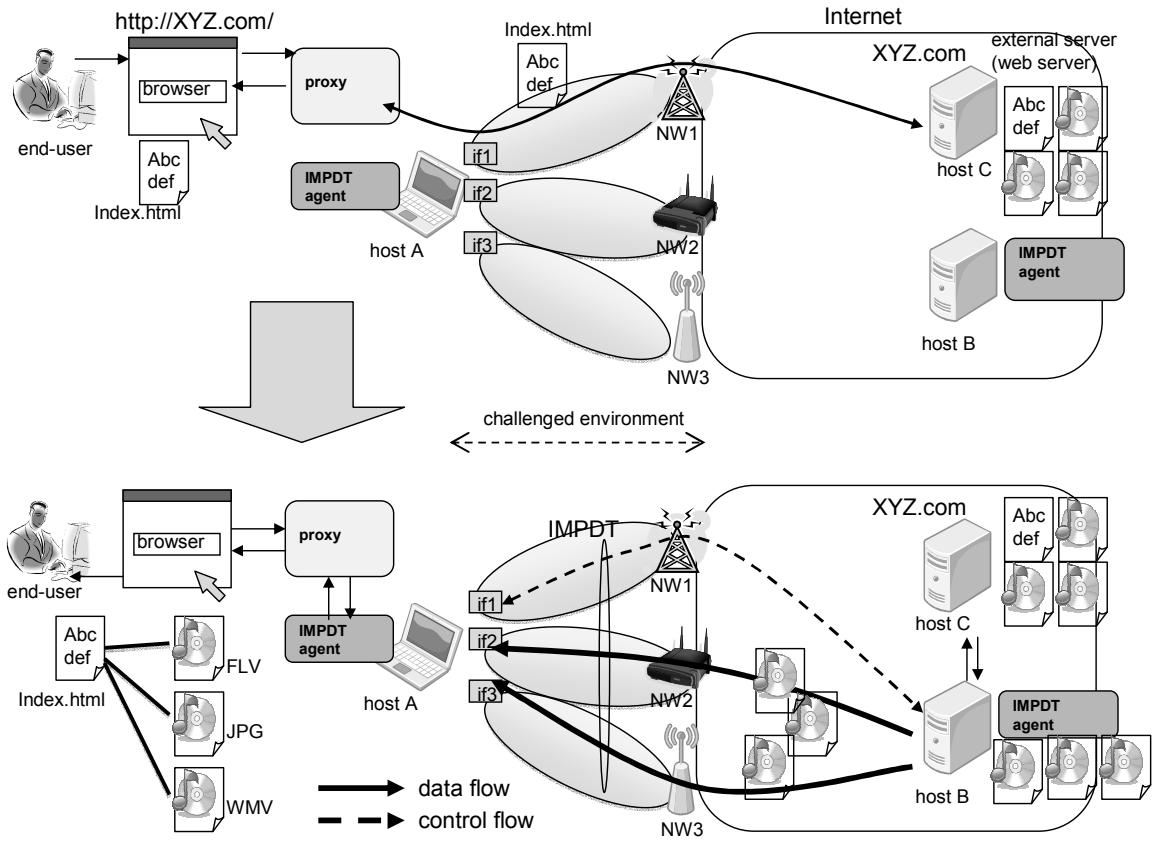


Figure 5.1: Overview of web access using IMPDT

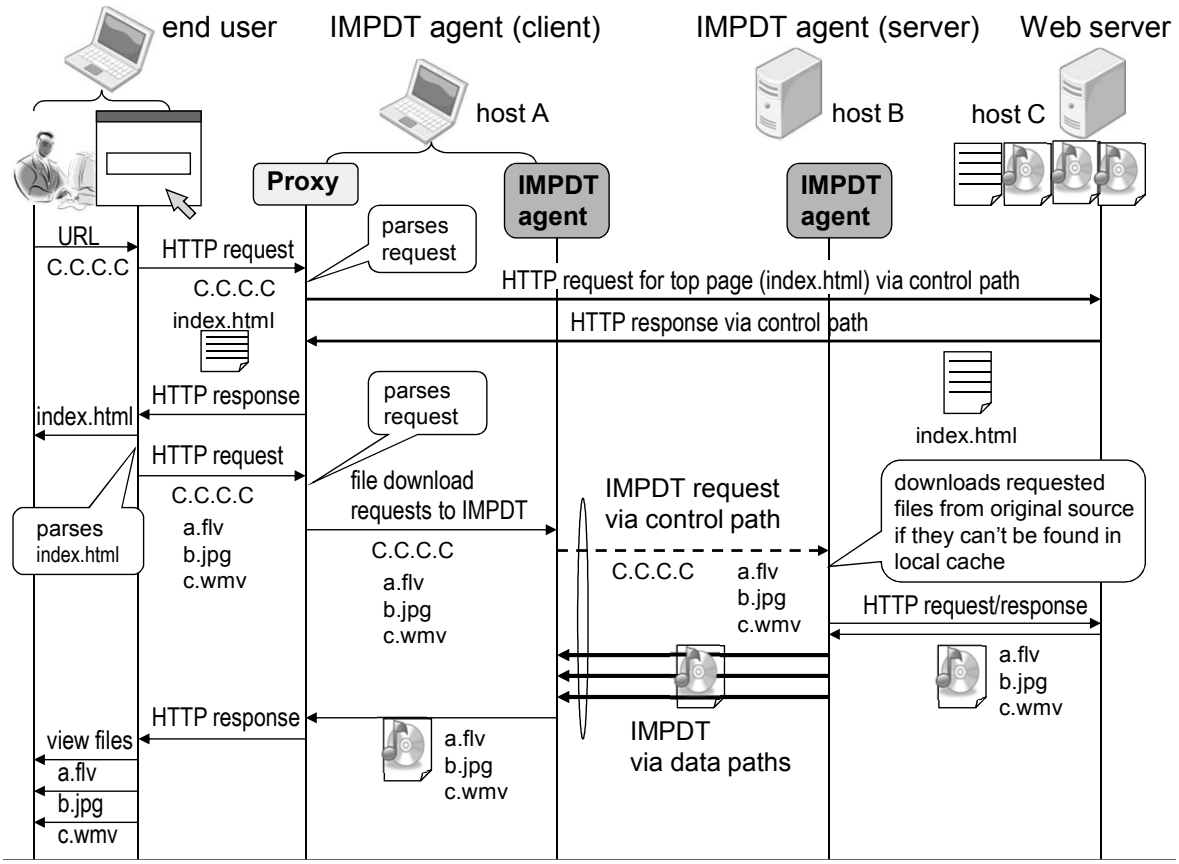


Figure 5.2: Example of control and data message sequence in web access using IMPDT

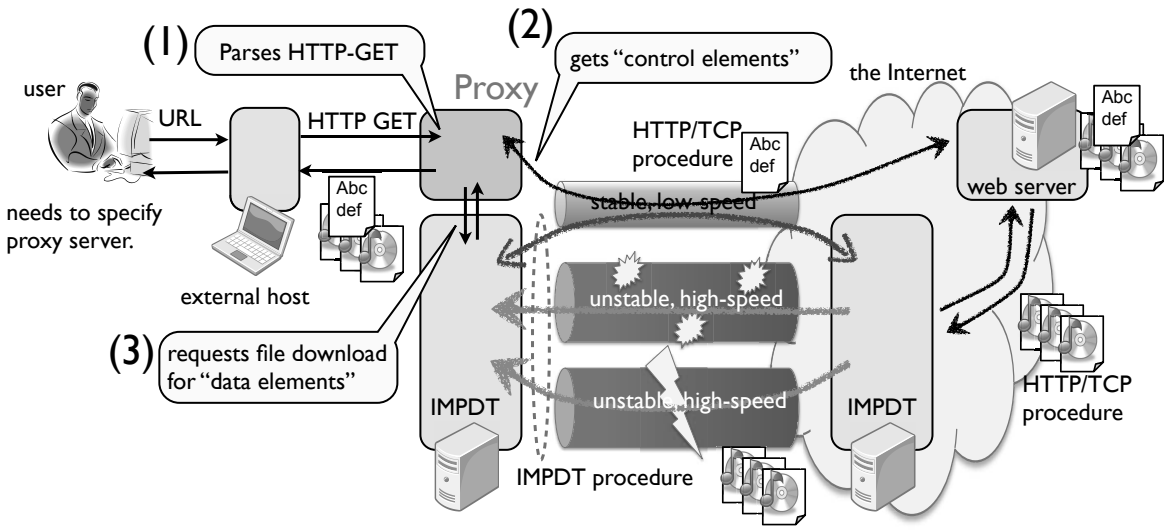


Figure 5.3: IMPDT as HTTP proxy

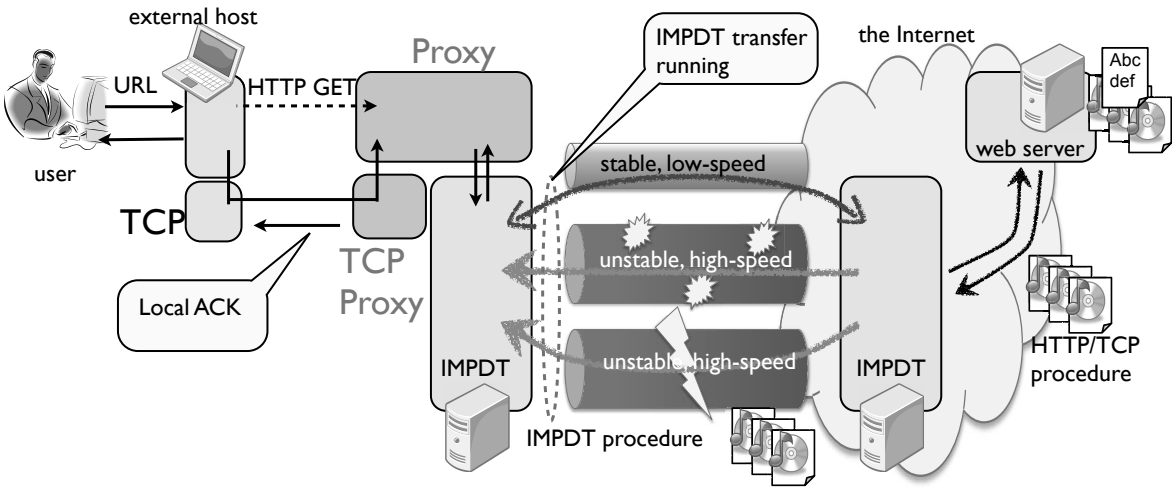


Figure 5.4: IMPDT as TCP proxy

# Chapter 6

# Delivering A File by Mutilpath-Multicast on OpenFlow Networks

## 6.1 Overview of Mutilpath-Multicast File Delivery

In this chapter, we focus on quick and efficient data transfer from a server in a wired backbone network to a mobile host in a wireless access environment. We discuss and propose a one-to-many file-delivery scheduling scheme, which we call a multipath multicast (MPMC) file transfer scheme, to enable a server to quickly and efficiently send a file to multiple wireless access base stations by the concurrent use of multiple available routes. The proposed scheme aims to (i) fully utilize the possible network bandwidth by transmitting segments (chunks) of a file simultaneously on multiple paths and (ii) reduce the bandwidth required to deliver a file to multiple receivers by multicasting the segments. On the basis of the maxflow problem, the scheme performs file segmentation, construction of multiple multicast trees, and segment transfer scheduling using multiple multicasts. Note that we refer to a server in a wired backbone network as a sender host and a wireless access base station as a receiver host in this chapter.

We use multiple multicast (MC) trees simultaneously to transfer a file quickly and ef-

ficiently. Note that file transfer completion times may vary among receivers. The central idea is to prioritize receiver with the best maxflow rate, and to construct multiple MC trees bound for (a part of) other receivers as leaves by extending the multiple paths to the preferred receiver. Once the preferred receiver has received the entire file, the receiver with the second fastest maxflow will be prioritized. File transfer continues in this manner until all receivers have received the entire file. In other words, this approach ensures maxflow to the preferred receiver at each stage and allows subsidiary data flows to other receivers simultaneously using residual available link capacities.

In this chapter, we report a multipath multicast (MPMC) file delivery scheduling scheme for a single source node sending a file to multiple receiver nodes based on the approach described above. The goal is to decrease file transfer time by aggregating the bandwidth of each sender-receiver path and by multicast transfer for efficient bandwidth utilization. A preliminary prototype system, which demonstrates the feasibility and usefulness of our proposal, was developed on OpenFlow networks. The reminder of this chapter is organized as follows. Sections 6.2 and 6.3 explain the model and algorithm of our MPMC file delivery scheduling scheme, respectively. We present a prototype of the scheme in Section 6.4. Evaluation experiment are discussed in Section 6.5, and conclusions are given in Section 6.6.

## 6.2 MPMC Model

We assume a sender $S$ and $N_R$ receivers $R_1, R_2, \ldots, R_{N_R}$. The graph $G(V, E)$ indicates a network that consists of all internal relay nodes. Sender $S$ is attached to $v_0 \in V$, and each receiver is attached to one relay node in $V$. Let $v_j^*$ denote the relay node to which $R_j$ is attached, and $V^*$ be the set of marginal attachment nodes, i.e., $\{v_0, v_1^*, \ldots, v_{N_R}^*\}$. Each unidirectional link $(v, v') \in E$ has maximum capacity $c(v, v')$ as a natural number, and $c(v, v') = c(v', v)$. Each link between a relay node and the sender or a receiver has infinite capacity. Figure 6.1 illustrates a network in which $V = \{v_0, v_1, v_2, v_3\}$ and $v_1^* = v_2, v_2^* = v_1$, and $v_3^* = v_3$. All directional links have equal capacity.

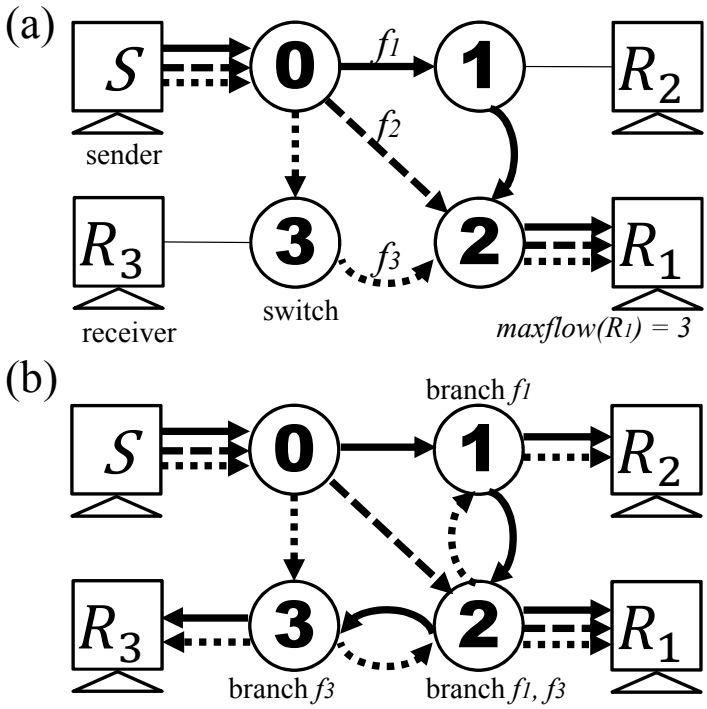


Figure 6.1: (a) A maxflow to $R_1$. (b) Multiple multicasting to $R_1, R_2, R_3$ based on the maxflow to $R_1$.

A multicast flow (MC-flow) is a data flow from sender to one or more receivers along a tree to transfer a part of a file. Note that a unicast flow (sender to a single receiver) is also treated as an MC-flow. Without loss of generality, each MC-flow consumes one (unit) capacity of each link traversed (i.e., the data rate of each MC-flow is one). A file is divided into equal segments (*chunks*), each of which is transferred by an MC-flow. A chunk may be transferred more than once by different MC-flows at different times for distribution to all receivers. In this model, the questions to be answered are as follows: how should a file be divided into chunks, what MC-flow should be used, how should the MC-flow be defined and used to transfer chunks, and how should the chunks be scheduled.

In general, each file transfer completion time to each receiver should be shortened. However, determining optimal scheduling is complex and finding the optimal solution is difficult.

Therefore, we do not define a global utility function to be maximized. Instead, first, we choose a feasible and reasonable solution that satisfies the necessary condition by a simple heuristic approach. This solution is not ensured to be optimal in any sense. The following approach is based on the maximum flow problem, which finds the maximum data rate (maxflow) $M_j$ from sender $S$ to a receiver $R_j$. We assign a higher priority to receiver $R_j$ with higher maxflow $M_j$ and search for a schedule that will complete the file transfer to $R_j$ as quickly as possible. Thereafter we search for a schedule for other lower priority receivers, based on that of the higher priority receiver.

## 6.3 MPMC Planning and Scheduling Algorithm

This section describes the procedure of MPMC planning and scheduling algorithm. First, for each $R_j$, we compute $M_j$. We sort and renumber all receivers in descending order of $M_j$; $R_1, R_2, \ldots, R_{N_R}$. Let $N_B$ be the least common measure of $\{M_j | j = 1, 2, \ldots, N_R\}$, and let us divide an entire file into $N_B$ equal chunks. Let $\mathbf{B_j}$ be the set of chunks that has been already received by $R_j$. Initially, $\mathbf{B_j}$ is set to $\emptyset$ for all $j$. Set $j = 1$ and then start. As shown in Figure 6.2, the procedure consists of repetitive four steps.

Step 1 Decompose maxflow $M_j$ to unit flows from $S$ to $R_j$.

$M_j$ can be realized by $M_j$ of unit flows $\{f_1, f_2, \ldots, f_{M_j}\}$, each of which runs on one particular network route (path) from $S$ to $R_j$ with the same unit data rate. Note that this decomposition to unit flows is not necessarily unique.

Step 2 Find a possible extension of each unit flow to an MC-flow from $S$ to some receivers.

Each unit flow $f$ can be branched and extended to form an MC-flow $f^*$. Thus, unit flows $f_1, f_2, \ldots, f_{M_j}$ are extended to MC-flows $f_1^*, f_2^*, \ldots, f_{M_j}^*$ according to the following constraints and criteria. Let $\mathbf{F}_0^V(v)$ and $\mathbf{F}_0^E(v, v')$

be the unit flow set (to $R_j$) traversing relay node $v$ and link $(v, v')$, respectively.

The MC-flow set $\mathbf{F}^E(v, v')$ traversing link $(v, v')$ is defined and constrained as follows. If $f \in \mathbf{F}_0^E(v, v')$, then $f^* \in \mathbf{F}^E(v, v')$. If $f \notin \mathbf{F}_0^E(v, v')$ and $f^* \in \mathbf{F}^E(v, v')$, then the three constraints hold.

- $f \notin \mathbf{F}_0^V(v')$, and $\forall w \neq v \left(f^* \notin \mathbf{F}^E(w, v')\right)$.
- $f \in \mathbf{F}_0^V(v)$, or $\exists w \left(f^* \in \mathbf{F}^E(w, v)\right)$.
- $c(v, v') \geq \left|\mathbf{F}^E(v, v')\right|$

Thereafter, the MC-flow set $\mathbf{F}^V(v)$ traversing relay node $v$ is defined as follows.

$$\mathbf{F}^V(v) = \mathbf{F}_0^V(v) \cup \left(\bigcup_{v'} \mathbf{F}^E(v', v)\right)$$

We then search for a possible extension to MC-flows according to the following criterion on MC-flow coverage to marginal relay nodes attached by non-preferred receivers. Let $V_j^* = V^* \setminus \{v_1^*, v_2^*, \ldots, v_j^*\}$ as the set of such marginal relay nodes.

- maximizing $\sum_{v \in V_j^*} \left|\mathbf{F}^V(v)\right|$,

where the obtained extension is not necessarily unique and is not ensured to be optimal.

In Fig. 6.1, the maxflows from $S$ to receivers $R_1, R_2, R_3$ are $M_1 = 3, M_2 = M_3 = 2$, respectively. Figure 6.1 (a) shows that three unit flows $f_1, f_2, f_3$ consist of a maxflow from $S$ to $R_1$. Figure 6.1 (b) shows how those unit flows are extended to three MC-flows: $f_1^*$ is to $\{R_1, R_2, R_3\}$, $f_2^*$ is to $\{R_1\}$ (not actually extended), and $f_3^*$ is to $\{R_1, R_2, R_3\}$, respectively, in the first round of chunk transfer.

Step 3 Assign each chunk to each MC-flow.

Each MC-flow $f_k^*$ ($k = 1, 2, \ldots, M_j$) is assigned to a chunk not received by $R_j$. A chunk transferred by $f^*$ is distributed not only to $R_j$ but also to $R_i$ if and only if $f^* \in \mathbf{F}^V(v_i^*)$ .

We search for a possible correspondence between $M_j$ MC-flows and $M_j$ chunks selected from $\mathbf{B_j}^c$ according to the following condition on the resulting (updated) sets $\{\mathbf{B}_{j+1}, \mathbf{B}_{j+2}, \ldots .\mathbf{B}_{N_R}\}$ of the set of chunks received by each non-preferred receiver.

- maximizing $\sum_{i=j+1}^{N_R} |\mathbf{B}_i|$ ,

where the obtained assignment is not necessarily unique and not ensured to be optimal. Step 3 corresponds to one round of chunk transfer by $M_j$ of MC-flows.

In Figure 6.1, the number of chunks is 6 (i.e., the LCM of $M_1, M_2, M_3$).

Step 4 Eliminate receivers that have received all chunks.

After updating each $\mathbf{B}_i$ ($i = j, j+1, \ldots, N_R$), if $\mathbf{B}_j$ does not include all chunks, return to Step 3 as the next round.

Otherwise, $j = j + 1$ until $\mathbf{B}_j$ includes all chunks.

If $j \leq N_R$, then return to Step 1 to obtain the new most preferred receiver, otherwise complete the procedure.

Figure 6.3 shows an example of MPMC planning and scheduling algorithm. A sender host $S$ sends a file to two receiver hosts $R_1, R_2$. M.F. denotes maxflow. $R_1$ is chosen as the most preferred receiver in the first round because it has a larger M.F. than that of $R_2$. Maxflow from $S$ to $R_1$ is decomposed to unit flows $A$, $B$ and $C$. To span $R_2$, flow $A$ and $B$ are branched at node 1 and 2, respectively. After chunk assignment, a file is divided into six chunks and

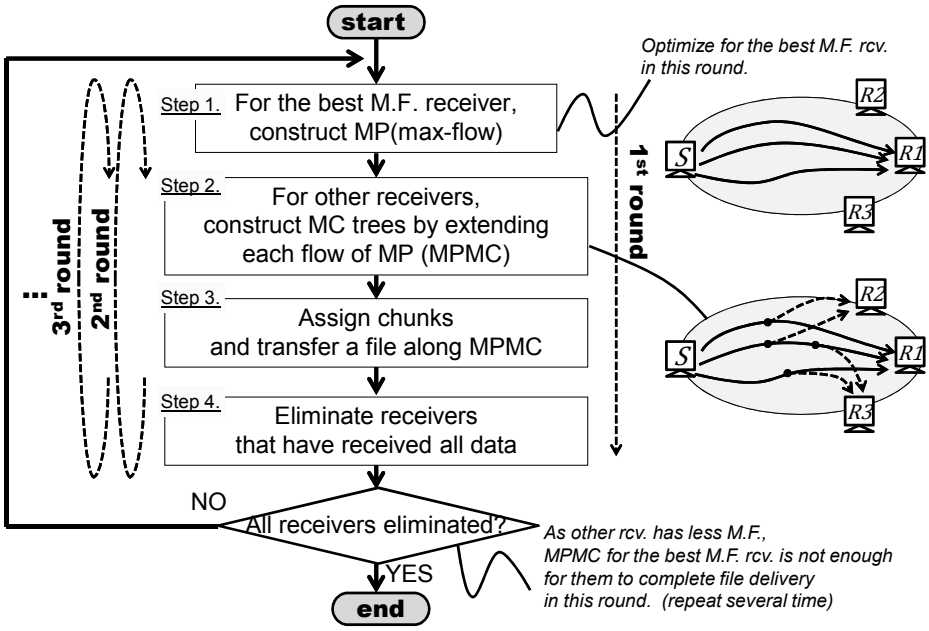


Figure 6.2: Basic idea of MPMC planning and scheduling algorithm

then chunks 1 and 2 are delivered along flow $B$, chunks 3 and 4 along flow $A$, and chunks 5 and 6 along flow $C$. As a result, $R_1$ receives all chunks in the first round and $R_2$ does chunks 1, 2, 3 and 4. $R_2$ is chosen as the most preferred receiver in the second round. Maxflow from $S$ to $R_2$ is similarly decomposed to unit flows $D$ and $E$, and then remaining chunks 5 and 6 are scheduled along these flows to be delivered from $S$ to $R_2$.

## 6.4 MPMC Prototype System

We developed a prototype MPMC system for file transfers on OpenFlow networks to examine the feasibility and usefulness of our proposal.

OpenFlow is a new standard for SDN. In contrast to conventional network routers/switches, OpenFlow separates the control and data planes. An OpenFlow controller (OFC) and OpenFlow switches (OFSs) are responsible for the control and data planes, respectively. The OFC

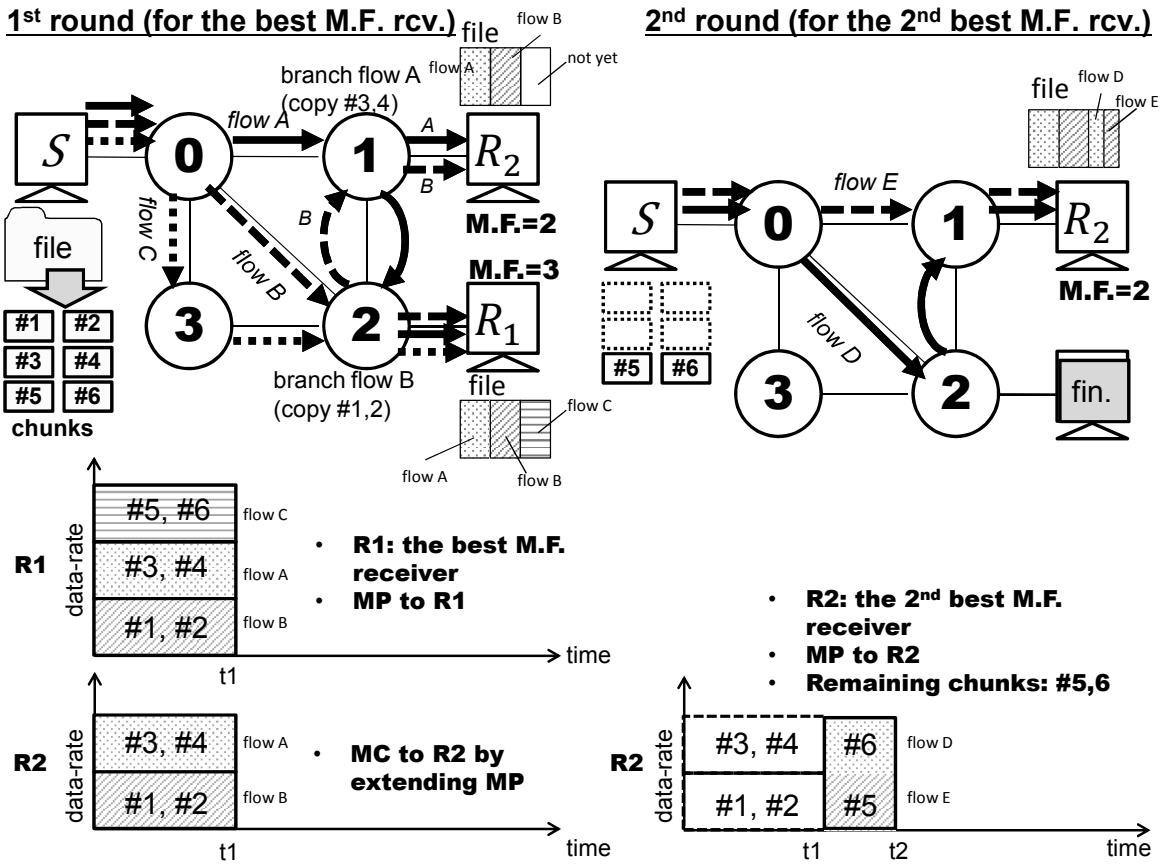


Figure 6.3: Example of MPMC planning and scheduling algorithm

dynamically installs and changes a flow table in each OFS. A flow table has three fields, packet header, action, and statistics, to define and manage the flows.

An MPMC controller performs MPMC file delivery planning. This includes optimal MPMC flow computation and chunk assignment along the flows. In addition, it configures each OFS to realize the flows from a sender to one or more receivers computed as per the algorithm. We implemented an MPMC controller in an OFC on the Trema framework [42]. A sender agent, implemented in the sender host, transfers chunks of a given file in cooperation with the MPMC controller. A receiver agent, implemented in each receiver host, joins the MC group related to this file transfer, receives each chunk transferred in the MPMC mode,

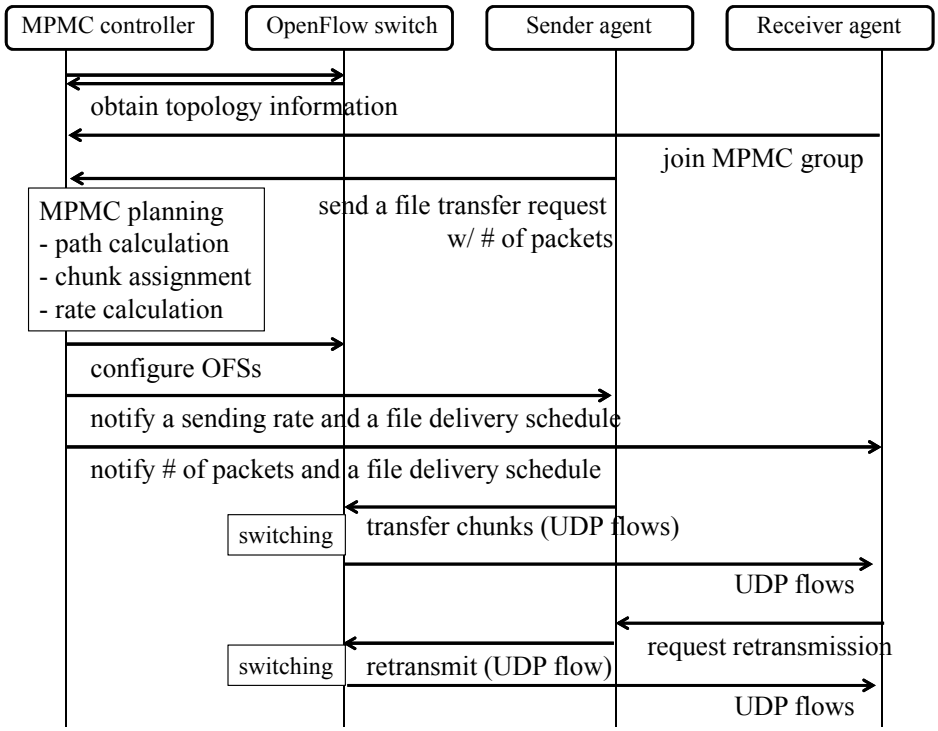


Figure 6.4: Basic procedure of proposed MPMC file delivery scheme

and assembles all chunks to retrieve the original file.

Figure 6.4 shows the basic procedure of the prototype system. The MPMC controller obtains topology information, including connectivity and link capacities between OFSs, in advance. The Link Layer Discovery Protocol provides link connectivity information to the OFC. The OFC obtains link speed information for active physical ports from all OFCs through the OpenFlow message exchange of Features Request/Reply, for simplicity, recognizes this link speed as link capacity (i.e., bandwidth). Note that this link speed value is configured via the Ethernet autonegotiation procedure and is not actual available bandwidth along the link between the OFSs.

A receiver agent sends Membership Report of the Internet Group Management Protocol as a joining request to MPMC file transfers. All OFSs are configured to forward this report to

the OFC (i.e., “packet-in” for OFC) when it is received; therefore, the OFC always receives this message via an OFS adjacent to the originating receiver agent. This allows the OFC to know which OFS a receiver agent is adjacent to.

A sender agent sends a file transfer request message to the OFC to start the file transfer. The OFC recognizes which OFS is adjacent to the sender agent in the same way adjacency is determined for a receiver agent. The sender agent sends the size of the given file to the OFC, which is determined by using the number of packets required to deliver a given file. Triggered by this notification, the MPMC controller performs the MPMC file delivery planning described in Section 6.3. The computed number of chunks and link bandwidth determine the maximum sending rate along each path. The OFC configures packet flow tables for each OFSs traversed by MPMC flows. Note that each chunk may consist of multiple packets and packets may be duplicated along these flows.

The sender agent uses the User Datagram Protocol (UDP) to send chunks to the network and generates the same number of UDP flows as that of chunks notified by the MPMC controller. Each UDP flow, which has different source port number, corresponds to each chunk and carries packets corresponding to each chunk. This rule allows the OFSs traversed by MPMC UDP flows to associate received UDP packets with their respective chunks by checking the source port number in the header. Thereafter, the OFS determines the output port for the UDP flow according to the flow table previously configured by the OFC. As mentioned previously, the flow table depends on MPMC planning.

The OFC transmits the MPMC planning result to the sender agent. The planning results include the number of chunks, number of receiver hosts, and file transfer schedule. The file transfer schedule determines when and which chunk (i.e., corresponding packets) is to be transferred and the maximum sending rate. The OFC notifies receiver agents of the total number of packets to be received, number of chunks, and file transfer schedule.

After the sender agent divides the file into chunks and divides each chunk into UDP packets, it transfers packets along with the file transfer schedule. The receiver agents receive

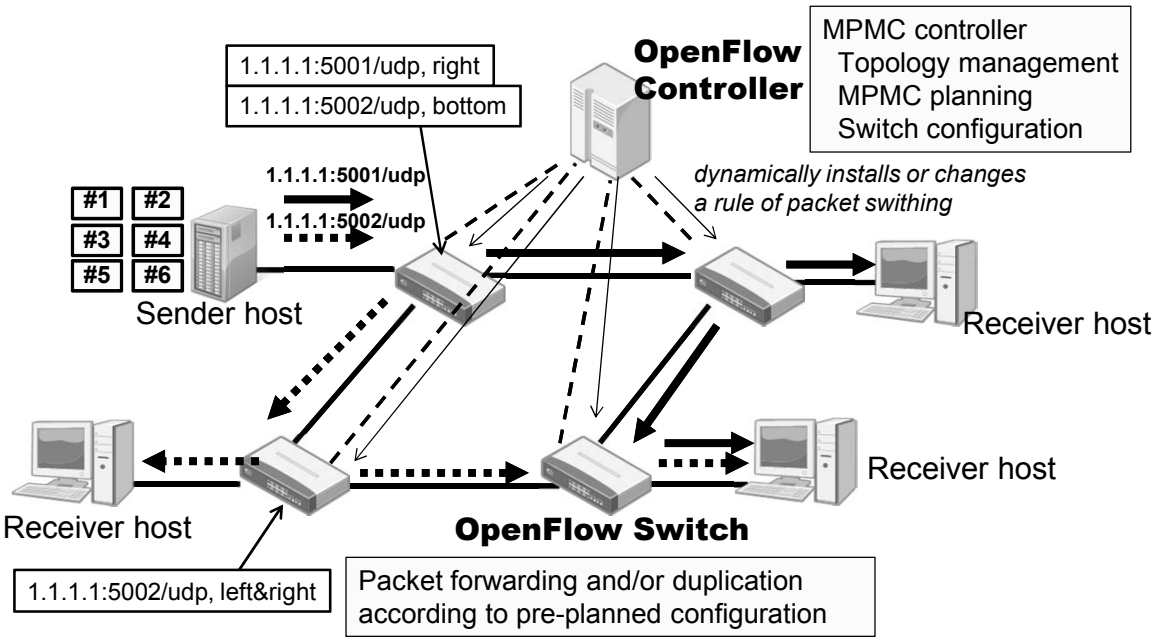


Figure 6.5: Overview of OpenFlow prototype system

packets and reassemble them to obtain the original file. When all packets have been received, the receiver agent notifies the sender agent that the entire file has been received.

Each UDP datagram has a unique sequence number that allows the receiver to know the correspondence between the original file and datagram. This sequence number is used by the receiver agent for reconstruction and packet loss recognition. This prototype system has easily implemented retransmission control. Packet loss is determined after the file transfer is completed. If there are lost packets, the receiver agent requests retransmission of those packets to the sender agent. Lost packets are identified by sequence number and are retransmitted by UDP unicast, i.e., independent of MPMC planning flow. The receiver agent recognizes the completion of transfer if and only if no packets are received in a pre-established period (1 s in this implementation).

## 6.5 Experiment and Discussion

To examine the feasibility and usefulness of the proposed MPMC file delivery, we conducted OpenFlow testbed experiments using the prototype system and compared the MPMC transmission completion time for each receiver with that of MC transmission.

A sender host $S$ transfers a file (50, 100, and 200 Mbytes) to multiple receiver hosts ($R_1, R_2, R_3$). Two network configurations are used: one is a same-link-capacity network, i.e., link capacity between OFSs is 100 Mbps, and the other is a mixed-link-capacity network that consists of 100 Mbps and 1 Gbps links. We examine the following three scenarios: 1) two receiver hosts in the same-link-capacity network, 2) two receiver hosts in the mixed-link-capacity network, and 3) three receiver hosts in the same-link-capacity network. Note that, for simplicity, in both same- and mixed-link-capacity networks, the link capacity between the end host and its adjacent OFS is 1 Gbps. The maximum sending rate for each flow is set by the OFC to fixed value determined by bottleneck link capacity.

### 6.5.1 Two Receivers in a Same-Link-Capacity Network

In scenario 1, sender host $S$ transfers a file in the MC and MPMC modes to receiver hosts $R_1$ and $R_2$ over the network shown in Figure 6.6 (a). For the MC mode, the flow capacity of $R_1$ and $R_2$ is 1. The maxflows of $R_1$ and $R_2$ are 3 and 2, respectively. As shown in Figure 6.7, the file is divided into six chunks (0–5) because the LCM of the maxflow of $R_1$ and that of $R_2$ is 6. These six chunks are allocated to three MPMC flows, i.e., two chunks per a flow. As $R_1$ receives three flows simultaneously, it receives all six chunks in the first round. As $R_2$ receives only two flows simultaneously, there are two unreceived chunks (4 and 5 in this case) after the first round. $R_2$ receives the remaining two chunks via two flows, i.e., one chunk per flow, in the second round. The average receiving rate of $R_1$ is 251 Mbps in MPMC, which is approximately three times higher than that of MC (91.5 Mbps). The average receiving rate of $R_2$ is 181 Mbps in MPMC, which is approximately two times higher than that of MC

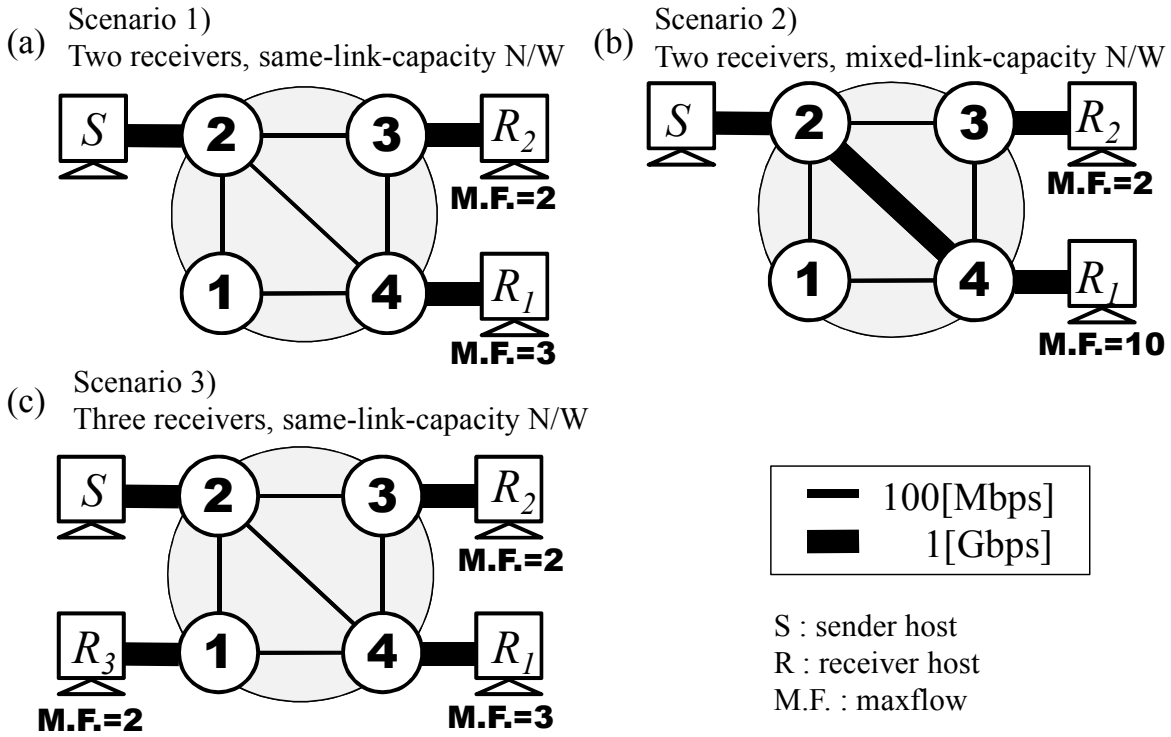


Figure 6.6: Network topology in experiments

(92.1 Mbps). As expected, these MPMC file deliveries achieve maxflow for each receiver host. We observed similar results for transmission completion times as shown in Figure 6.8 and Table 6.2.

### 6.5.2 Two Receivers in a Mixed-Link-Capacity Network

In scenario 2, sender host $S$ transfers a file in the MC and MPMC modes to receiver hosts $R_1$ and $R_2$ over the network shown in Figure 6.6 (b). For the MC mode, the flow capacity of $R_1$ and $R_2$ is 1. The maxflows of $R_1$ and $R_2$ are 12 and 2, respectively. As shown in Figure 6.9, the file is divided into twelve chunks (0–11). $R_1$ receives twelve chunks along with its maxflow in the first round and achieves a much higher receiving rate than that of $R_2$. Figure 6.10 and Table 6.3 show the transmission completion time results. As shown

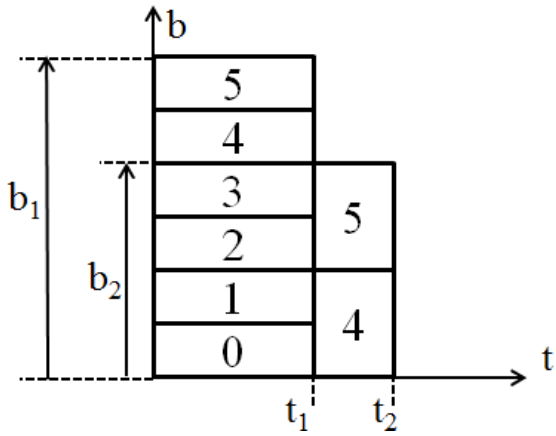


Figure 6.7: MPMC file transfer schedule (Scenario 1)

Table 6.1: Flow capacity and receiving rate (Scenario 1)

| | MC | | MPMC | |
|---|---|---|---|---|
| | $R_1$ | $R_2$ | $R_1$ | $R_2$ |
| Flow capacity | 1 | 1 | 3 | 2 |
| Rcv. rate[Mbps] | 91.5 | 92.1 | 251 | 181 |

in Table 6.4, the average receiving rate of $R_1$ in the MPMC mode is 706 Mbps. This is expected to be approximately twelve times higher than that of MC mode (90.2 Mbps), while it is approximately only eight times. This is because the link capacity between the sender host and its adjacent OFS is the bottleneck link capacity, which is less than the maxflow of $R_1$, and the sender host's socket processing overhead cannot be ignored in such a high-speed transfer.

### 6.5.3 Three Receivers in a Same-Link-Capacity Network

In scenario 3, sender host $S$ transfers a file in the MC and MPMC modes to receiver hosts $R_1, R_2$ and $R_3$ over the network shown in Figure 6.6 (c). For the MC mode, the flow capacity

Table 6.2: Transmission completion time (Scenario 1)

| File size | MC | | MPMC | |
|---|---|---|---|---|
| [MBytes] | $R_1$[s] | $R_2$[s] | $R_1$[s] | $R_2$[s] |
| 50 | 4.42 | 4.43 | 1.66 | 2.24 |
| 100 | 9.03 | 8.85 | 3.19 | 4.45 |
| 200 | 18.3 | 18.3 | 6.67 | 9.50 |

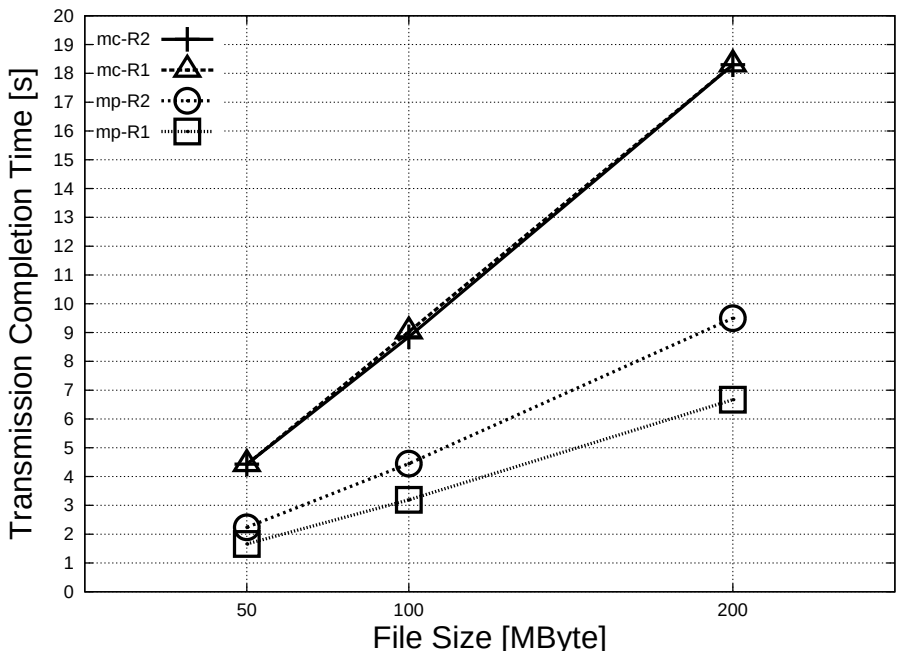


Figure 6.8: Transmission completion time (Scenario 1)

of $R_1, R_2$, and $R_3$ is 1. The maxflows of $R_1, R_2$, and $R_3$ are 3, 2, and 2, respectively. Figure 6.11 and Table 6.5 show the transmission completion time results. As shown in Table 6.6, the average receiving rates of $R_1, R_2$, and $R_3$ in the MPMC mode are 252, 156, and 122 Mbps, respectively. All average MPMC receiving rates are much higher than the MC rates (approximately 92 Mbps). However, in the MPMC mode, $R_2$ and $R_3$ have different average

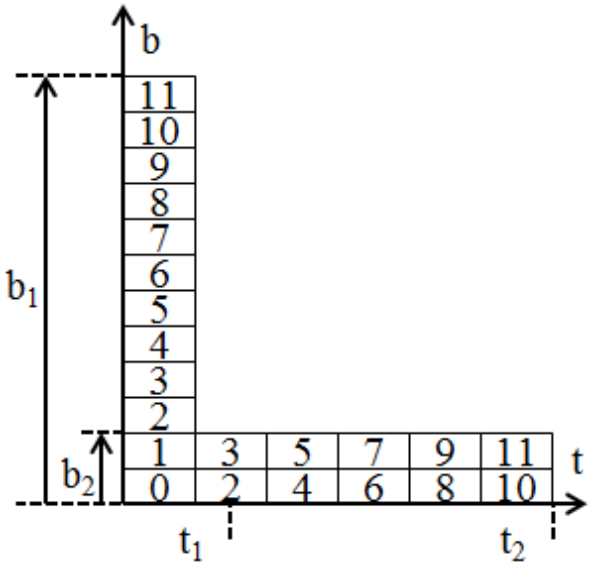


Figure 6.9: MPMC file transfer schedule (Scenario 2)

receiving rates despite having the same maxflows.

Figures 6.12 and 6.13(a) show the details of the file transfer schedule for the MPMC mode in a same-link-capacity network with three receivers. A file is divided into six chunks and $R_1$ receives the complete file in the first round. Although $R_2$ completes in the second round, $R_3$ requires an additional round. However, as shown in Figure 6.13(b), an optimal scheduling exists in which all hosts receive the complete file by the second round. This

Table 6.3: Transmission completion time(Scenario 2)

| File size | MC | | MPMC | |
|---|---|---|---|---|
| [MBytes] | $R_1$[s] | $R_2$[s] | $R_1$[s] | $R_2$[s] |
| 50 | 4.58 | 4.58 | 0.87 | 2.41 |
| 100 | 9.15 | 9.15 | 1.06 | 4.76 |
| 200 | 18.28 | 18.28 | 1.90 | 9.45 |

Table 6.4: Flow capacity and receiving rate (Scenario 2)

| | MC | | MPMC | |
|---|---|---|---|---|
| | $R_1$ | $R_2$ | $R_1$ | $R_2$ |
| Flow capacity | 1 | 1 | 12 | 2 |
| Rcv. rate[Mbps] | 90.2 | 90.2 | 706 | 173 |

reveals that our current implementation does not provide an optimal MPMC scheduling. This may result from a lack of considerations on flow extension and chunk assignments for lower maxflow receivers. Further improvements are required.

Table 6.5: Transmission completion time (Scenario 3)

| File size | MC | | | MPMC | | |
|---|---|---|---|---|---|---|
| [MBytes] | $R_1$[s] | $R_2$[s] | $R_3$[s] | $R_1$[s] | $R_2$[s] | $R_3$[s] |
| 50 | 4.42 | 4.43 | 4.47 | 1.64 | 2.60 | 3.35 |
| 100 | 8.99 | 8.84 | 8.94 | 3.20 | 5.17 | 6.70 |
| 200 | 18.3 | 18.3 | 18.3 | 6.71 | 11.0 | 13.8 |

Table 6.6: Flow capacity and receiving rate (Scenario 3)

| | MC | | | MPMC | | |
|---|---|---|---|---|---|---|
| | $R_1$ | $R_2$ | $R_3$ | $R_1$ | $R_2$ | $R_3$ |
| Flow capacity | 1 | 1 | 1 | 3 | 2 | 2 |
| Rcv. rate[Mbps] | 91.7 | 92.1 | 91.6 | 252 | 156 | 122 |

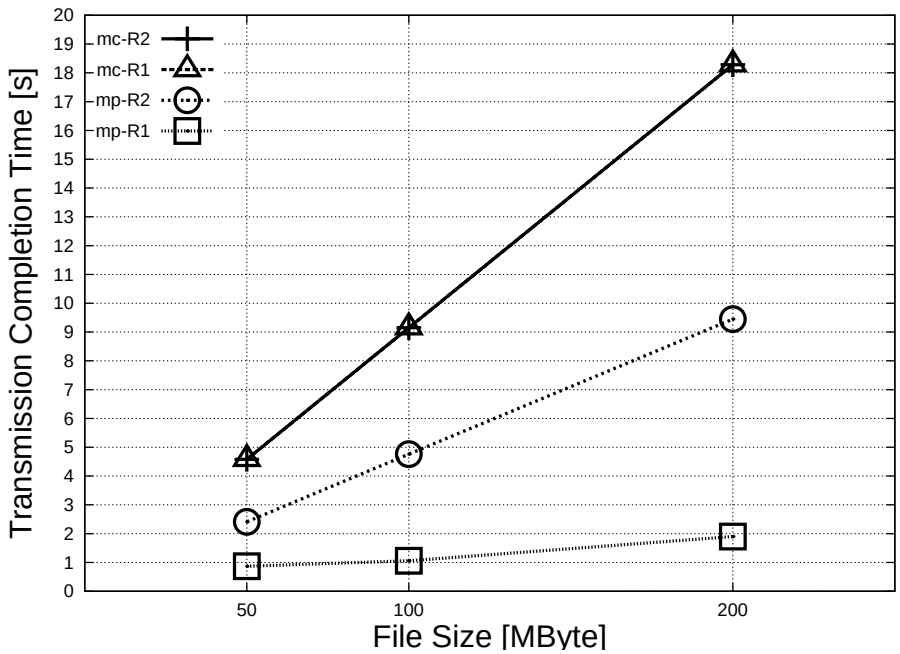


Figure 6.10: Transmission completion time (Scenario 2)

## 6.6 Conclusion

We proposed an MPMC file transfer scheme that contributes to reduced transmission times for one-to-many file transfers. This contribution is realized by (i) fully utilizing the possible network bandwidth by transmitting segments of a file simultaneously on multiple paths and (ii) reducing the bandwidth required to deliver a file to multiple receivers by multicasting. We developed a prototype MPMC system for file transfers on OpenFlow networks and experimentally examined its feasibility and usefulness in three scenarios. For simplicity, the current prototype system was implemented using preliminary retransmission control and did not contain adaptive rate control. MPMC planning should be improved to determine the optimal transfer solution. We should also examine and cope with the scalability issue in practical use. In a large scale network with heterogeneous links, the LCM of maxflows of all receivers, i.e., the number of segments, could become large, leading to increased computa-

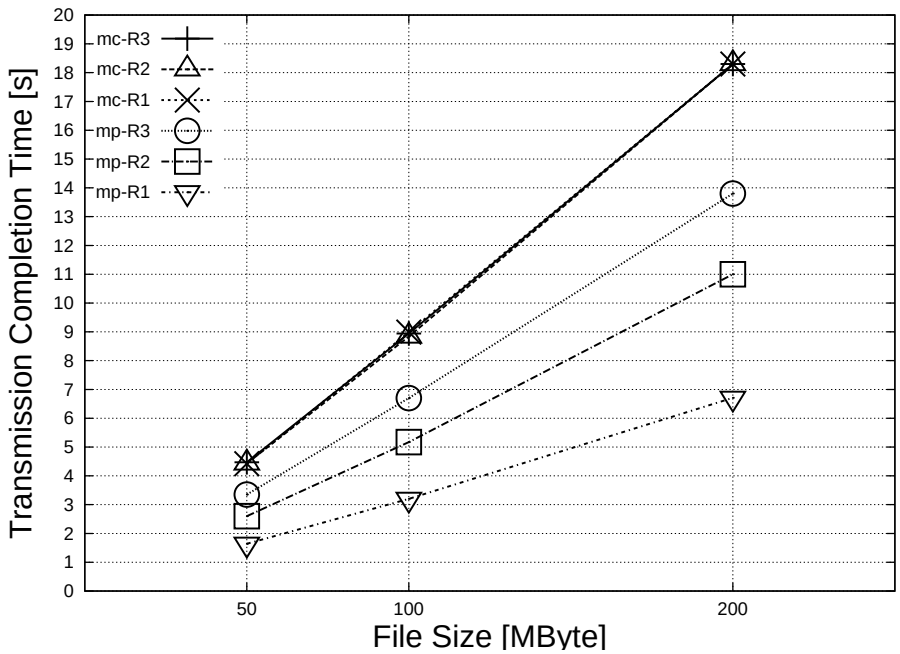


Figure 6.11: Transmission completion time (Scenario 3)

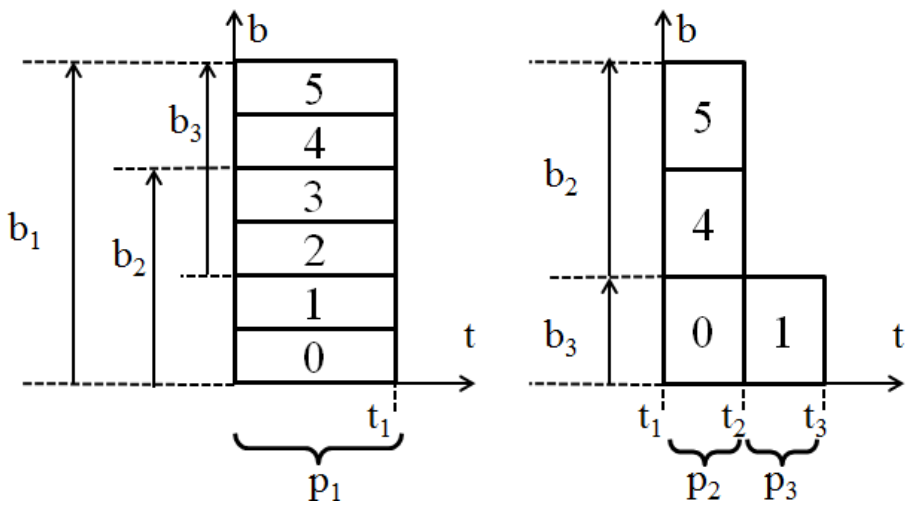


Figure 6.12: MPMC file transfer schedule (Scenario 3)

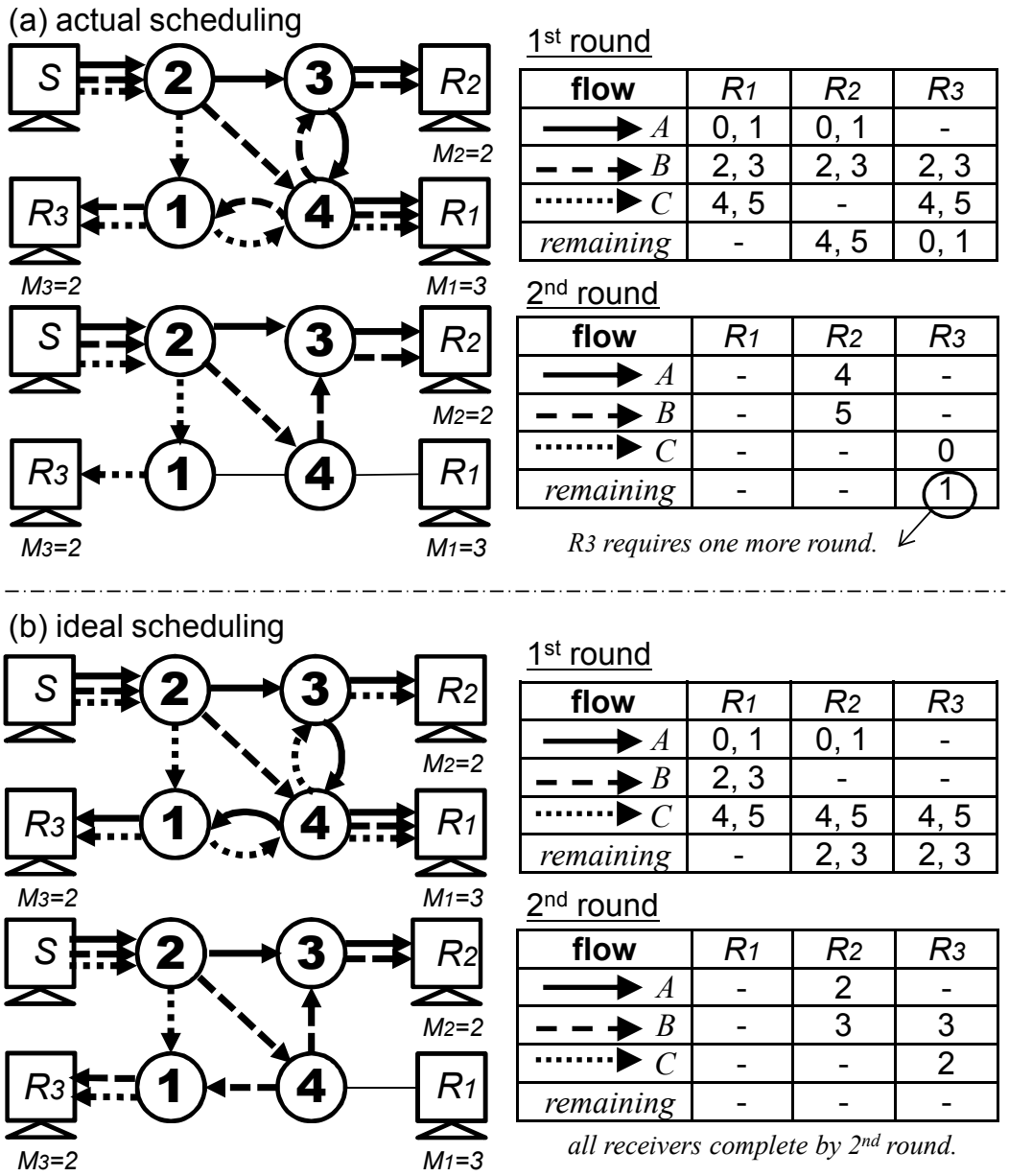


Figure 6.13: The detail of MPMC file delivery schedule (Scenario 3)

tional and processing times.

# Chapter 7

# Concluding Remarks

In this dissertation, in order to realize efficient data transfer in a challenged network environment that does not meet application requirements, we focused on a novel data transfer framework that enables the concurrent and integrated use of heterogeneous multiple networks.

In Chapter 3, we introduced the concept of integrating multipath data transfer (IMPDT) to efficiently transfer a large packet of data if multiple heterogeneous networks are available, but when none of them has sufficient performance for the requested task. IMPDT integrates heterogeneous networks not simply for bandwidth aggregation but also for providing sustainable control information exchange on a stable but low data rate network path, being handled separately from data transmission on different network paths in different ways. We discussed the design of a proof-of-concept implementation for IMPDT framework, including the mechanisms of retransmission, the rate adjustment of each data flow, and the data-flow setup control. We examined the basic performance of the prototype system in an indoor experiment using a simulated network. The evaluation results proved that the system effectively utilized the bandwidth of multiple data paths by aggregation.

In Chapter 4, we conducted three field experiments using a combination of terrestrial communication links and a satellite communication link in order to validate our prototype

implementation in a real-world communication services environment. The first experiment used a combination of 3G ,Wi-Fi, and a satellite communication link (ETS-VIII). ETS-VIII link was regarded as a slow but stable network and better suited for control flow in that experiment. Our prototype implementation performed better than a simple aggregation of transfer by parallel TCP connections. It clearly indicated the potential that the basic concept of a steady extra control channel could improve transfer performance in challenged environments. In the second experiment, a commercial global satellite communication network was used as a stable but low data rate link for emergency networking in disaster fields. The results proved that our system worked as expected—even in a moving vehicle environment. The third experiment used a pair of a high-speed satellite communication links: (WINDS) and 3G. The WINDS link was regarded as having a high data rate but being transiently lossy and a disrupted link in that experiment. The results validated that a separated control flow via a slower but more stable 3G network enables the transfer of large packets of data by exploiting a high-speed WINDS link even under lossy and/or disrupted conditions, i.e., unstable conditions. Through these experiments, we also clarified some problems in a real-world communication service environment and improved our system.

In Chapter 5, we designed and implemented a prototype web access system for web browsing over multiple challenged networks by integrating the HTTP protocol and the IMPDT scheme. Our system handles each file consisting of a web page differently and selects an appropriate transport-layer method for the file. We also discussed a case in which our system functions as a TCP proxy to interwork smoothly with internal and external web browsers.

In Chapter 6, we proposed a multipath multicast (MPMC) file-transfer scheme that contributes to reduce transmission times for one-to-many file transfers from a server to multiple wireless access base stations. This contribution was realized by: (i) fully utilizing the possible network bandwidth by transmitting segments of a file simultaneously on multiple paths, and, (ii) reducing the bandwidth required to deliver a file to multiple receivers by multicasting. We developed a prototype MPMC system for file transfers on OpenFlow networks and

experimentally examined its feasibility and usefulness in three scenarios.

# Bibliography

[1] A. Nagata, S. Yamamura, M. Uchida and M. Tsuru, "Proxy Data Transfer System with a Stable Control Channel and Dynamically Changing Data Channels," Proceeding of the 3rd ACM Workshop on Challenged Networks (CHANTS'08), pp.121–124, September 2008. (DOI: 10.1145/1409985.1410009)

[2] A. Nagata, S. Yamamura, M. Tsuru, "Integrating Multiple and Heterogeneous Challenged Networks for Large-Sized Data Transfer," Proceeding of International Conferene on Intelligent Networking and Collaborative Systems (INCoS 2009), pp.263–264, Nov.2009. (DOI 10.1109/INCOS.2009.71)

[3] A. Nagata, S. Yamamura, M. Tsuru, "Integrating Multiple and Heterogeneous Challenged Networks – Detailed Design and Experimental Evaluation using Satellite Link –," Proceeding of the 2nd International Conference on Intelligent Networking and Collaborating Systems (INCOS), pp.362–367, November 2010. (DOI: 10.1109/INCOS.2010.48)

[4] A. Nagata, S. Yamamura, M. Tsuru, "Virtual Single Network Path by Integrating Multiple and Heterogeneous Challenged Networks," IEICE Transactions on Communications, vol. E94-B, no. 6, pp.1546–1555, June 2011. (DOI: 10.1587/transcom.E94.B.1546)

[5] A. Nagata, S. Yamamura, and M. Tsuru, "Web Browsing over Multiple Heterogeneous Challenged Networks," Proceeding of the 4th ACM Workshop on Challenged Networks (CHANTS'11). pp. 57–60, September 2011. (DOI: 10.1145/2030652.2030669)

[6] A. Nagata, S. Yamamura, M. Tsuru, "Data transfer exploiting multiple heterogeneous challenged networks - implementation and application," Int. J. Space-Based and Situated Computing, No.2, Vol.2, pp.112–122, June 2012. (DOI: 10.1504/IJSSC.2012.047468)

[7] A. Nagata, Y. Tsukiji, and M. Tsuru, "Delivering A File by Multipath-Multicast on OpenFlow networks," Proceeding of the 5th International Workshop on Information Network Design (WIND'13), pp.8350–840, September 2013. (DOI 10.1109/INCoS.2013.160)

[8] M. Stemm, and R. H. Katz, "Vertical handoffs in wireless overlay networks," Mobile Networks and Applications, vol. 3, issue 4, pp. 335–350, December 1998.

[9] W. T. Chen, J. C. Liu, and H. K. Huang, "An adaptive scheme for vertical handoff in wireless overlay networks," Proceedings of 10th International Conference on Parallel and Distributed Systems (ICPADS 2004), pp. 541–548, July 2004.

[10] K. Tsukamoto, Y. Fukuda, Y. Hori, and Y. Oie, "New TCP Congestion Control Schemes for Multimodal Mobile Hosts," IEICE Transactions on Communications, Vol.E89-B, No.6, pp.1825–1836, June 2006.

[11] J. Roy, V. Vaidehi, and S. Srikanth, "Always best-connected QoS integration model for the WLAN, WiMAX heterogeneous network," Proceeding of the 1st International Conference on Industrial and Information Systems (ICIIS), pp. 361–366. August 8–11, 2006. Peradeniya, Sri Lanka.

[12] R. Stewart, Q. Xie, K. Morneault, C. Sharp, H. Schwarzbauer, T. Taylor, I. Rytina, M. Kalla, L. Zhang, and V. Paxson, "Stream Control Transmission Protocol," RFC2960, Internet Engineering Task Force (IETF), 2000.

[13] E. Kohler, S. Floyd, M. Handley, and J. Padhye, "Datagram Congestion Control Protocol (DCCP)," RFC4340, Internet Engineering Task Force (IETF), March 2006.

[14] J. R. Iyengar, P. D. Amer, and R. Stewart, "Concurrent multipath transfer using SCTP multihoming over independent end-to-end paths," IEEE/ACM Transactions on Networking, vol. 14, no. 5, pp. 951–964. October 2006.

[15] H. Han, S. Shakkottai, C. V. Hollot, R. Srikant, D. Towsley, "Multi-Path TCP: A Joint Congestion Control and Routing Scheme to Exploit Path Diversity in the Internet," IEEE/ACM Transactions on Networking, vol. 14, no. 6, pp.1260–1271. December 2006.

[16] K. Fall, "A delay-tolerant network architecture for challenged internets," Proceedings of ACM SIGCOMM '03, pp.27–34. August 25–29, 2003.

[17] IRTF DTNRG, Delay Tolerant Networking Research Group. Obtained through the Internet: *http://www.dtnrg.org/*

[18] S. Farrell, and V. Cahill, "Evaluating LTP-T: A DTN-Friendly Transport Protocol," Proceeding of the 3rd International Workshop on Satellite and Space Communications (IWSSC '07), pp.178–181, September 2007.

[19] S. Burleigh, M. Ramadas, and S. Farrell, "Licklider Transmission Protocol - Motivation," RFC 5325, Internet Engineering Task Force (IETF), September 2008.

[20] P. Kyasanur, J. Padhye, and P. Bahl, "On the efficacy of separating control and data into different frequency bands," Proceeding of the 2nd International Conference on Broadband Networks (BROADNETS), vol. 1, pp.602–611. October 2005.

[21] A. Balasubramanian, B. Levine, and A. Venkataramani, “DTN Routing as a Resource Allocation Problem,” Proceedings of ACM SIGCOMM 2007, pp.373–384. August 2007.

[22] N. Banerjee, M. D. Corner, and B. N. Levine, “An Energy-Efficient Architecture for DTN Throwboxes,” IEEE INFOCOM 2007, pp.776–784, May 2007.

[23] S. Jain, M. Demmer, R. Patra, and K. Fall, “Using redundancy to cope with failures in a delay tolerant network,” Proceedings of ACM SIGCOMM 2005, pp.109–120. August 22–26, 2005.

[24] R. Mahajan, J. Padhye, R. Raghavendra, and B. Zill, “Eat All You Can in an All-You-Can-Eat Buffet: A Case for Aggressive Resource Usage,” Paper Presented at the 7th ACM Workshop on Hot Topics in Networks (HotNets-VII), pp.43–48. October 6–7, 2008.

[25] H. Burton, and D. Sullivan, “Error and error control,” Proceedings of the IEEE, vol. 60, no. 11, pp.1293–1301. November 1972.

[26] S. McCanne, V. Jacobson and M. Vetterli, “Receiver-driven layered multicast,” Proceeding of ACM SIGCOMM ’96, 117–130.

[27] A. Legout and E. W. Biersack, “PLM: Fast Convergence for Cumulative Layered Multicast Transmission Schemes,” ACM SIGMETRICS 2000, 13–22.

[28] T. Miyoshi and Y. Tanaka, “Adaptive Content Delivery System with Multicasting and Buffering for Heterogeneous Networks,” IEICE transactions on information and systems E88-D(2), 204–213, 2005.

[29] Y. Cui, X. Wang, H. Wang, G. Pan, and Y. Wang, “FMTCP: A Fountain Code-Based Multipath Transmission Control Protocol,” IEEE 32nd International Conference on Distributed Computing Systems (ICDCS ’12), 366–375.

[30] T. Nogichi, T. Matsuda, and M. Yamamoto, “Performance Evaluation of New Multicast Architecture with Network Coding,” IEICE transactions on communications E86-B(6), 1788-1795, June 2003.

[31] T. Matsuda and T. Takine, “Multicast Communications with Reed Solomon/Network Joint Coding in Wireless Multihop Networks,” Journal of Communications, Vol. 4, No. 11, pp. 856-864, Dec. 2009.

[32] JAXA, “Engineering Test Satellite VIII ”KIKU No.8” (ETS-VIII),” Obtained through the Internet: *http://www.jaxa.jp/projects/sat/ets8/index_e.html*

[33] I. Rhee and L. Xu, “CUBIC: A new TCP-friendly high-speed TCP variant,” Proceedings of Protocols for Future, Large-Scale and Diverse Network Transports (PFLDNeT), 2005.

[34] L. A. Grieco and S. Mascolo, “Performance Evaluation and Comparison of Westwood+, New Reno, and Vegas TCP Congestion Control,” ACM Computer Communication Review, Vol. 34(2), April 2004

[35] JAXA, “Wideband InterNetworking engineering test and Demonstration Satellite ”KIZUNA” (WINDS),” Obtained through the Internet: *http://www.jaxa.jp/projects/sat/winds/index_e.html*,

[36] D. Asahizawa, Y. Sato, G. Sato, and Y. Shibata, “Disaster Surveillance Video Transmission System by Wireless Ballooned Network,” Proceeding of the 23rd IEEE International Conference on Advanced Information Networking and Applications Workshops (WAINA ’09), pp. 884–889. May 26–29, 2009. Bradford, UK.

[37] K. Scott, and S. Burleigh, “Bundle Protocol Specification,” RFC5050, Internet Engineering Task Force (IETF), November 2007.

[38] J. Ott, and D. Kutscher, “A Disconnection-Tolerant Transport for Drive-thru Internet Environments,” Proceedings of IEEE INFOCOM 2005, vol. 3, pp.1849–1862. March 13–17, 2005.

[39] J. Ott, and D. Kutscher, “Bundling the Web: HTTP over DTN,” Paper Presented at WNEPT Workshop on Networking in Public Transport, 9 pages, August 2006.

[40] J. Border, M. Kojo, J. Griner, G. Montenegro, and Z. Shelby, “Performance Enhancing Proxies Intended to Mitigate Link-Related Degradations,” RFC3135, Internet Engineering Task Force (IETF), June 2001.

[41] “OpenFlow,” Obtained through the Internet: *http://www.openflow.org/*

[42] “Trema,” Obtained through the Internet: *http://trema.github.com/trema/*