Concurrent Multipath Transmission to Improve Performance for Multi-homed Devices in Heterogeneous Networks

Allen Lehopotseng Ramaboli

This thesis is submitted in fulfillment of the academic requirements
for the degree of
Doctor of Philosophy in Electrical Engineering
in the Faculty of Engineering and The Built Environment
University of Cape Town
February 2016

Supervisor

Associate Professor **Olabisi E. Falowo**

Co-supervisor

Professor **H. Anthony Chan**

Declaration

I hereby declare that: (1) the above thesis is my own unaided work, both in conception and execution, and that apart from the normal guidance of my supervisor, I have received no assistance apart from that stated below; (2) except as stated below, neither the substance or any part of the thesis has been submitted in the past, or is being, or is to be submitted for a degree in the University or any other University.

I am now presenting the thesis for examination for the Degree of Doctor of Philosophy in Electrical Engineering. I also grant the University free license to reproduce the above thesis in whole or in part, for the purpose of research.

ALLEN LEHOPOTSENG RAMABOLI 15/02/2016

Name Date

To my family for their patience and love

Abstract

Recent network technology developments have led to the emergence of a variety of access network technologies—such as IEEE 802.11, wireless local area network (WLAN), IEEE 802.16, Worldwide Interoperability for Microwave Access (WIMAX) and Long Term Evolution (LTE)—which can be integrated to offer ubiquitous access in a heterogeneous network environment. User's devices also come equipped with multiple network interfaces to connect to the different network technologies, making it possible to establish multiple network paths between end hosts. However, the current connectivity settings confine the user's devices to using a single network path at a time, leading to low utilization of the resources in a heterogeneous network and poor performance for demanding applications, such as high definition video streaming.

The simultaneous use of multiple network interfaces, also called bandwidth aggregation, can increase application throughput and reduce the packets' end-to-end delays. However, multiple independent paths often have heterogeneous characteristics in terms of offered bandwidth, latency and loss rate, making it challenging to achieve efficient bandwidth aggregation. For instance, striping the flow's packets over multiple network paths with different latencies can cause packet reordering, which can significantly degrade performance of the current transport protocols. This thesis proposes three new solutions to mitigate the effects of network path heterogeneity on the performance of various concurrent multipath transmission settings.

First, a network layer solution is proposed to stripe packets of delay-sensitive and high-bandwidth applications for concurrent transmission across multiple network paths. The solution leverages the paths' latency heterogeneity to reduce packet reordering, leading to minimal reordering delay, which improves performance of delay-sensitive applications. Second, multipath video streaming is developed for H.264 scalable video, where the reference video packets are adaptively assigned to low loss network paths to reduce drifting errors, thus combatting H.264 video distortion effectively. Finally, a new segment scheduling framework—which carefully considers path heterogeneity—is incorporated into the IETF Multipath TCP to improve throughput performance. The proposed solutions have been validated using a series of simulation experiments. The results reveal that the proposed solutions can enable efficient bandwidth aggregation for concurrent multipath transmission over heterogeneous network paths.

Acknowledgements

I express my greatest gratitude to my supervisor, Associate Professor Olabisi Falowo, for the patience and knowledge in guiding my effort to complete this research. Working with Professor. Falowo has taught me that hard work, discipline and integrity are key to embarking on challenging and eventually fruitful research. His commitment to excellence has greatly shaped this work.

I also express my appreciation to Professor H. A. Chan for his kind assistance along my research path.

I thank the Communications Research Group for the funding that helped me to present my research work at some of the best conferences.

Last but not least, I thank family and friends for their encouragement and support. They have all been the pillar of my research endeavor.

Table of Contents

List of Figures

List of Tables

Glossary

Packet Reordering: an event where the higher sequence number packet arrives at the receiver before the lower sequence packet.

Reorder Entropy (RE): measures the disorder in the received packet sequence

Reorder Density (RD): measures the amount and extent of packet reordering in a sequence of packets arriving at the receiver

Reorder Buffer Density (RBD): measures buffer occupancy frequencies normalized to the number of non-duplicate packets in the arriving packet sequence

Bandwidth Aggregation: pooling bandwidth from multiple network paths to create a larger logical path that a multi-homed mobile device can use for concurrent multipath transmission

Concurrent Multipath Transmission (CMT): striping packets from the same flow over multiple network paths for parallel transmission

Multi-homing: the process of allowing a device with multiple network interfaces to be addressable through more than one of the interfaces

Re-sequencing-Unit (RU): entity that reassembles packets received from multiple network interfaces and tries to resolve packet reordering

Reordering Delay: the time a reordered packet takes in the reorder buffer until a lower sequence number packet arrives to resolve the packet reordering

Multipath Packet Scheduler (MPS): a scheduling mechanism that decides how to adaptively stripe packets simultaneously across multiple network paths

Video Packets Distributor (VPD): a scheduler to assign video packets to multiple network paths for concurrent multipath video streaming

Scalable Video Coding (SVC): encoding of a video sequence into base layer and several enhancement layers

Group of Pictures (GoP): specifies the decodable structure of a group of video frames

Real-time Transport Protocol (RTP): protocol to deliver real-time data over IP network

Real-time Control Transport Protocol (RTCP): collects performance reports on the transmission of RTP data

Multipath-RTP: RTP over multiple network paths

Real-time Streaming Protocol (RTSP): used for establishing and controlling media sessions

RTCP-based Path Manager: monitors end-to-end network paths characteristics using the RTCP reports

Session Manager: establishes and manages video sessions in concurrent multipath transmission

Multi-homed Device: a device with multiple network interfaces that are addressable through different IP networks

Peak Signal to Noise Ratio (PSNR): measures video quality in decibels on a logarithmic scale using mean square error (MSE) between the reference and the received video sequences with the square of the highest sample value in a video image, which is 255 for 8-bit image

Structural Similarity Index (SSIM): an objective video quality metric that measures structural similarities between two image sequences in terms of luminance, contrast and spatial texture

Decodable Frame Ratio (DFR): measures the percentage of decodable video frames

Bandwidth Asymmetry: different bandwidth settings for the network paths in multipath environment

Delay Heterogeneity: different delays between the network paths in multipath transmission

Loss Heterogeneity: different loss rates between the network paths in multipath transmission

Network Abstraction Layer Unit (NALU): defines the encapsulation of the coded video data

Base Layer (BL): the basic layer of H.264 encoded video, which does not depend on other layers for decoding

Enhancement Layer (EL): video layer to enhance video quality but depends on lower video layers for decoding

MPTCP Sub-flow: TCP instance in multipath TCP

Chapter 1

Introduction

The Internet traffic continues to grow at an alarming rate. Several phenomena—such as social networks, telecommuting and online gaming with high bandwidth multimedia content—contribute significantly to the growth of the Internet traffic. Cisco predicts that the Internet traffic, mostly consumed by mobile users, will see a 10-fold increase between 2014 and 2019 [1]. This rapid growth in Internet traffic makes it challenging for network operators to scale up capacity accordingly. This is despite the recent network technology developments that have led to the emergence of a variety of access network technologies, such as IEEE802.11—the wireless local area network (WLAN) standard, IEEE802.16—the Worldwide Interoperability for Microwave Access (WiMAX) and Long Term Evolution (LTE). At best, network operators can currently deploy these network technologies in a heterogeneous environment to offer ubiquitous access for users with mobile devices that are equipped with multiple network interfaces, thus allowing for network connectivity to be maintained while moving across a range of wireless network technologies. Access ubiquity, however, may not result in significant jump in data rates, and additional solutions are, therefore, required to improve data rates for emerging high-bandwidth applications. Concurrent multipath transmission has recently been proposed as one of the solutions to improve data rates for demanding applications [2] [3].

Concurrent multipath transmission allows for bandwidth units from multiple network interfaces on a communication device to be aggregated into a single logical capacity that has enough bandwidth to meet the capacity needs of 'bandwidth-hungry' applications. Bandwidth aggregation for concurrent multipath transmission becomes especially necessary when no single network path has enough bandwidth to meet the quality of service (QoS) requirements of the application. Concurrent multipath transmission, therefore, can provide a unique ability for network operators to scale up network capacity to meet high-bandwidth demands and enhance performance in terms of increased application throughput and reduced latency [4, 5, 6]. To benefit from a concurrent multipath transmission setup, traffic from a high-bandwidth application can be striped appropriately across the device's multiple network interfaces for concurrent transmission to a

destination device that is equipped with the corresponding network interfaces, which virtually increases the amount of bandwidth available to the application. Another way to reap the benefits of multipath transmission is to duplicate traffic across multiple network interfaces to improve resilience. This thesis focuses on the development and evaluation of efficient mechanisms to stripe application traffic across multiple network interfaces simultaneously to improve performance.

1.1 The Problem

A scenario for concurrent multipath transmission is illustrated in Fig. 1.1. The user's device connects to the network using two network interfaces, thus making it possible to stripe application traffic simultaneously across the two network interfaces to realize concurrent multipath transmission. A user's device that connects to more than one network simultaneously (as in Fig. 1.1) is said to be multi-homed [7]. However, the mere ability of a user's device to use multiple network paths at the same time may not be enough to efficiently utilize the bandwidth capacity that has been aggregated from the different networks. This is because the heterogeneous delays, bandwidth offerings and loss characteristics of the different network paths can result in data transmission anomalies, such as packet reordering and load imbalance, which can significantly degrade application performance in terms of throughput. Packet reordering occurs when higher sequence number packets arrive at the receiver earlier than lower sequence number packets. Recovery from packet reordering can impose undesirable delays on time-sensitive applications, and for TCP applications, spurious retransmissions are often triggered [8][9].

To address the problem of packet reordering, appropriate bandwidth aggregation solutions, aiming at distributing application traffic across multiple network paths and ensuring that data packets from the same application arrive at the receiver in correct order must be developed. Besides solving the packet reordering problem, tackling load imbalance is also critical to accomplishing efficient bandwidth aggregation for concurrent multipath transmission. Furthermore, some of the high bandwidth applications, such as multilayer encoded video, may benefit from application specific concurrent multipath transmission optimizations. This thesis attempts to provide solutions to the challenges of concurrent multipath transmission. The result is an adaptive concurrent multipath transmission framework, which can improve performance for demanding applications. The work is, therefore, an important step towards making bandwidth aggregation for concurrent multipath transmission a profitable solution for network operators and users in the Future Internet.

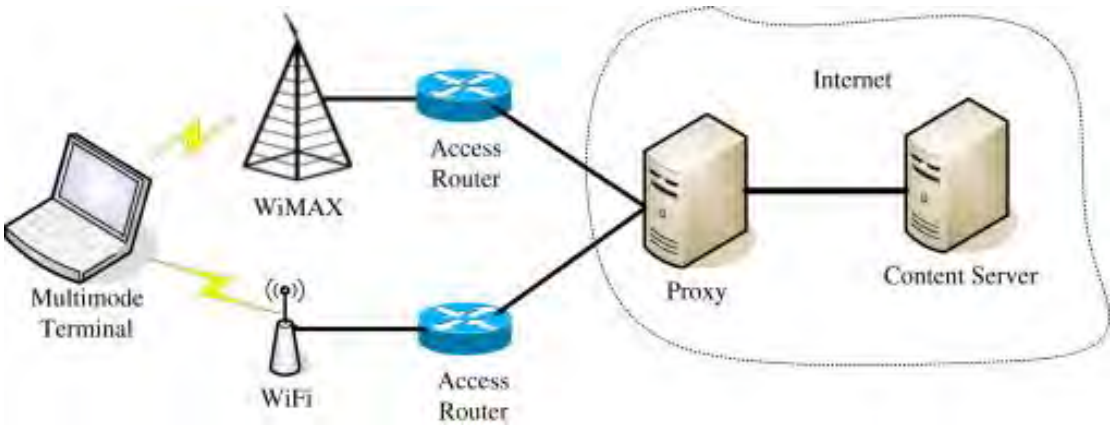

Fig. 1.1 Architecture supporting concurrent multipath transmission

1.2 Research Objectives

The research in this thesis examines bandwidth aggregation for concurrent multipath transmission where the application data can be striped across the device's multiple network interfaces for improved performance. The main goal of the research is to develop and evaluate efficient traffic allocation and distribution mechanisms to adaptively schedule packets from the same application across multiple network paths for concurrent transmission, ensuring that the packets arrive at the receiver in correct order. Specific objectives of the research are outlined as follows:

- To carry out a comprehensive study of recent research on bandwidth aggregation for concurrent multipath transmission. The result of the study is a critical review of various approaches that have recently been proposed to enable concurrent multipath transmission in heterogeneous networks.
- To devise a network-layer bandwidth aggregation framework for concurrent multipath transmission and then develop an efficient scheduling mechanism to combat the problem of packet reordering.
- To develop an application-specific concurrent multipath transmission system, which aims at providing H.264 SVC video-on-demand service for multi-homed user's devices in a heterogeneous network.
- To improve the throughput performance of the IETF Multipath TCP by proposing

a new scheduling mechanism, which is robust to path heterogeneity in terms of loss and delay.

- To design and develop suitable simulation experiments to evaluate the behavior of the proposed concurrent multipath transmission solutions.
- To present and critically analyze the experimental results to establish the efficacy of the proposed concurrent multipath transmission mechanisms.

1.3 Research Scope and Limitations

Concurrent multipath transmission can be used to realize redundancy by duplicating application packets across multiple network paths, and it can also be used to improve performance by striping application packets simultaneously across several network paths with heterogeneousl8 path characteristics. The scope of the research is limited to the latter concurrent multipath transmission use case, which attempts to improve network resource utilization to enhance performance of high bandwidth applications in heterogeneous networks. The application-specific concurrent multipath transmission system proposed in the thesis is limited to H.264 SVC video-on-demand streaming but the main ideas may be applied to transmit other video formats over multiple heterogeneous network paths. The traffic distribution mechanisms developed in this thesis enable concurrent multipath transmission mostly in the downlink but they can be adapted and applied in the uplink. The experimental method in the research is limited to simulation using an event-driven simulator such as the ns-3. To evaluate the proposed H.264 SVC multipath streaming approach, several real video traces have been used to generate traffic for the ns-3 simulation runs. The user terminals discussed in the thesis are mobile devices with multiple network interfaces but the effect of mobility of the devices on concurrent multipath transmission is not investigated in this thesis.

1.4 Summary of Research Contributions

The research presented in this thesis has resulted in several new contributions to knowledge in the area of bandwidth aggregation for concurrent multipath transmission. To assure the validity of the contributions made, some peer-reviewed publications have been produced as outlined in section 1.6. The key contributions of this research are summarized as follows:

1. The first contribution is a comprehensive and critical review of bandwidth aggregation solutions for concurrent multipath transmission in heterogeneous networks. The review provides useful information for research in bandwidth aggregation, identifying, comparing and classifying various solutions from previous research in the area. The end result of the critical review is a discussion of open research questions that need to be answered to advance knowledge in bandwidth aggregation for concurrent multipath transmission. The contribution is presented in Chapter 3. A peer-reviewed publication in the Elsevier Journal of Network and Computer Applications is evidence of this contribution. The journal publication has received over 40 citations.
2. The second contribution, which is presented in Chapter 4, is a network layer framework to enable concurrent multipath transmission. The framework includes a novel packet scheduling mechanism to stripe packets from the same application across multiple heterogeneous network paths simultaneously. The main purpose of the scheduling mechanism is to combat packet reordering by distributing the application's packets across the different network paths taking into consideration the paths' delay characteristics. Several packet reordering metrics have been used to evaluate the efficacy of proposed scheduling mechanism. The contribution has been published in part in the 2013 IEEE Conference on Advanced Information Networking and Applications.
3. The third contribution (Chapter 5) is a bandwidth aggregation framework for multipath streaming of H.264 scalable video coding (SVC) in heterogeneous networks. The proposed streaming solution improves H.264 SVC video-on-demand delivery by distributing the SVC packets across multiple network paths according to the packets' importance. The most important video packets such as the base layer packets are scheduled for transmission on the network paths with the lowest loss rates to minimize video distortion. The experimental framework to validate the proposed multipath video streaming solution is based on the ns-3 simulation environment and real video traces with varying characteristics. The video traces were processed using the Joint Scalable Video Model software. The results of this work have been published in the 2013 IEEE Conference on Military Communications and a journal article has also been submitted for publication in the IEEE Transactions on Consumer Electronics.

4. The fourth and final contribution, which is presented in Chapter 6, is the development of a scheduling framework to improve performance of the IETF Multipath TCP. The proposed scheduling solution addresses the short-comings of the default MPTCP segment schedulers. The solution ensures that MPTCP can realize and sustain bandwidth aggregation benefit under various path heterogeneity conditions. This work has been submitted for review in the IEEE Transactions on Consumer Electronics.

1.5 Thesis Organization

The rest of the thesis is organized into the following chapters:

Chapter 2 presents an overview of bandwidth aggregation for concurrent multipath transmission in heterogeneous networks. Important components supporting bandwidth aggregation for concurrent multipath transmission in heterogeneous networks are discussed. The benefits of exploiting bandwidth aggregation for concurrent multipath transmission are presented. Also, major challenges that can impede efficient bandwidth aggregation are highlighted. Recognizing that one of the most important objectives of a bandwidth aggregation solution is to address the problem of packet reordering, some of the important metrics to measure the extent of the packet reordering problem are introduced. Chapter 3 provides a comprehensive and critical review of the previous bandwidth aggregation solutions in heterogeneous networks; particularly, the focus is on protocols, traffic scheduling and distribution schemes developed to exploit the aggregated bandwidth capacity for concurrent multipath transmission. The aim of the review is to shed light on the state-of-the-art research on bandwidth aggregation in heterogeneous networks.

Chapter 4 introduces the proposed bandwidth aggregation framework for concurrent multipath transmission at the network layer. The framework includes a new packet scheduling mechanism developed to mitigate packet reordering to improve performance for real-time applications. The proposed scheduling mechanism adapts to variations in estimated end-to-end delay between the different network paths that form part of the concurrent multipath transmission system. Chapter 5 discusses a new video streaming system that enables the transmission of H.264 SVC over multiple paths for multi-homed user's devices in a heterogeneous network environment. The proposed multipath streaming system works to prevent loss of important (reference) video packets by striping such packets over the least loss paths. For recovery from pak2cket reordering, the multipath video streaming solution makes use of the packet scheduler developed in chapter 4.

In chapter 6, the IETF Multipath TCP is introduced and the default schedulers, which are responsible for distributing segments across the MPTCP sub-flows are highlighted. A new scheduling system that improves on the default schedulers is proposed and evaluated in a simulation environment. Chapter 7 summarizes the thesis and highlights important areas of future work to further improve concurrent multipath transmission in heterogeneous networks.

1.6 List of Publications

1. **Bandwidth Aggregation in Heterogeneous Wireless Networks: A Survey of Current Approaches and Issues** Elsevier Journal of Network and Computer Applications, November 3, 2012
   Authors: A. L. Ramaboli, O. E. Falowo, H. A. Chan
2. **Improving H.264 Scalable Video Delivery for Consumer Communication Devices with Multiple Network Interfaces** IEEE Transactions on Consumer Electronics [submitted]
   Authors: A. L. Ramaboli, O. E. Falowo, H. A. Chan
3. **Enhancing Multipath TCP to Improve Performance for Multi-homed Consumer Devices** IEEE Transactions on Consumer Electronics [submitted]
   Authors: A. L. Ramaboli, O. E. Falowo, H. A. Chan
4. **Using Multiple Links Simultaneously to Increase Capacity for Multi-homed Terminals in Heterogeneous Wireless Networks** IEEE AINA 2013 March 25, 2013
   Authors: A. L. Ramaboli, O. E. Falowo, H. A. Chan
5. **Improving H.264 Scalable Video Delivery for Multi-homed Terminals Using Multiple Links in Heterogeneous Wireless Networks** IEEE MILCOM 2013 November 7, 2013
   Authors: Allen Ramaboli, O. E. Falowo, H. A. Chan

Chapter 2

Overview of Bandwidth Aggregation for Concurrent Multipath Transmission in Heterogeneous Networks

This chapter provides an overview of bandwidth aggregation, which forms the basis for accomplishing concurrent multipath transmission for multi-homed devices in a heterogeneous network environment. The relation between multi-homing and bandwidth aggregation for concurrent multipath transmission is discussed. Then, the benefits and some of the challenges impeding the efficacy of bandwidth aggregation solutions are presented. The major components that constitute a bandwidth aggregation system for concurrent multipath transmission are described. Finally, the performance metrics that have been proposed to quantify packet reordering—one of the major challenges of concurrent multipath transmission—are introduced.

2.1 Bandwidth aggregation and multi-homing

Concurrent multipath transmission, which allows application traffic to be split across several network paths at the same time, has gained increasing research interest in academia and industry. Concurrent multipath transmission is made possible by multi-homing [10] and bandwidth aggregation in a heterogeneous network environment. A heterogeneous network environment, as illustrated in Fig. 2.1, comprises two or more independent networks that are somewhat integrated to improve the users' service experience. Different ways to integrate different networks can be found in [11]. The users' communication devices that are equipped with more than one network interface can connect to several networks simultaneously in a heterogeneous network. When this happens, each of the network interfaces on the user's device is independently addressable and the user's device is consequently considered multi-homed.

Bandwidth aggregation ideas build on the foundations of multi-homing, allowing for bandwidth from multiple network interfaces of a multi-homed user's device to be carefully harvested and aggregated creating a high-bandwidth logical link. The resulting high-bandwidth link can then be used by a multi-homed device for applications with high-bandwidth demands. The multi-homing configuration discussed in the research goes beyond the basic configuration, which

only allows for the use of a single network path at a time, switching to the next network path only in the event of failure and/or degraded performance. The research considers multi-homing where any number of network paths can be used simultaneously for data transmission, which effectively enables concurrent multipath transmission.

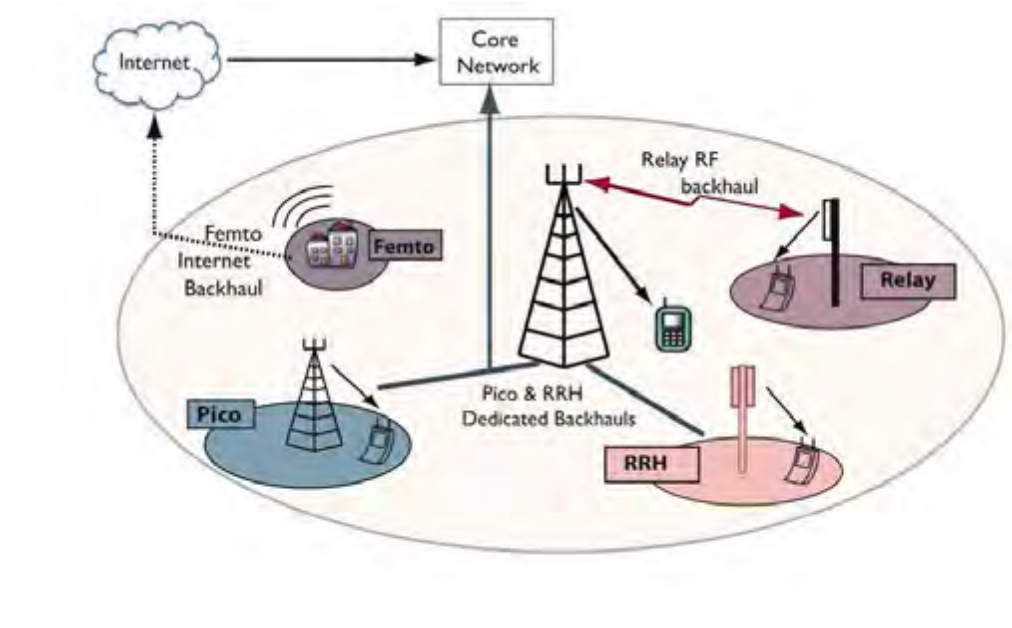

Fig. 2.1 A heterogeneous wireless network

2.2 Benefits of Bandwidth Aggregation

Aggregating effective offered bandwidth from individual network interfaces (network paths) to create a logical link with higher bandwidth has several benefits to both the network operator and the subscribers that have devices that are equipped with multiple network interfaces. The benefits are discussed below.

2.2.1 Increased Throughput

In a heterogeneous network, when an application requires a higher throughput than individual low-bandwidth network paths can provide, the bandwidth offered by several network paths that the user's device can connect to can be aggregated to create a larger logical link that is expected to offer enough bandwidth to meet the desired throughput guarantees. For instance,

consider two network paths: one network path offering 100kbps and the other providing 80kbps of bandwidth. Aggregating the bandwidth offerings from the network paths results in 180kpbs of total bandwidth. The application can exploit the aggregated bandwidth capacity by having its traffic striped simultaneously over the network paths for concurrent multipath transmission. This has been shown to yield throughput as high as the sum of throughput of the individual network paths [12]. Also, end-to-end delay can be effectively minimized as a direct result of larger aggregated bandwidth capacity [13][14], and this creates an opportunity to boost performance of delay-sensitive applications.

2.2.2 Improved Packet Delivery and Reliability

Besides performance guarantees in terms of throughput and delay, bandwidth aggregation can be used to improve packet delivery and reliability. That is, data packets can be duplicated and transmitted over multiple network paths to the same destination. When copies of the same packet arrive at the receiver, the error-free copy will be used, while the erroneous one will be discarded. In the event the packet copies all arrive in error, error correction and frame combining schemes can be used to resolve the errors, thus improving packet delivery [15]. Reliability can be accomplished by maintaining redundant network paths so that data traffic can be switched to unused network paths when the active network path fails.

2.2.3 Load Balancing

The ability of a user's device to use its multiple network interfaces simultaneously can help to ease load on one particular network interface by appropriately dispersing traffic over several other network interfaces. The load can be distributed evenly or unevenly among the available paths depending on the dynamics of traffic and network conditions as dictated by the type of traffic scheduling mechanism in place. When balancing traffic load over multiple network paths for concurrent transmission, the traffic distribution policy used should not compromise correct packet sequence as that can lead to degraded performance of the concurrent multipath transmission system. Load balancing over multiple network paths can be performed at flow level or packet level [16]. Flow level load balancing confines packets of the same flow to the same network path for the duration of the session or until the network path fails. This can lead to poor load balancing

when traffic flows have different sizes. However, packet reordering, which is a major concern of concurrent multipath transmission, can be evaded since packets of the same flow mostly travel the same path in a first-in first-out (FIFO) manner.

Packet level load balancing, on the other hand, allows for packets of the same flow to traverse different paths to the same destination, thus ensuring more efficiently balanced traffic load and preventing the formation of packet clusters in the network [17]. However, packet level load balancing over multiple network paths that have heterogeneous delay characteristics is more likely to cause high packet reordering, which is not desirable in concurrent multipath transmission. As it was highlighted earlier, high packet reordering can lead to poor performance of TCP and real-time applications.

2.2.4 Low-cost Capacity Increase

When network capacity is to be increased, options include but not limited to purchasing and deploying additional network infrastructure, and aggregating two or more of the low-bandwidth links to create a larger logical link that can be used to handle high bandwidth demands. The latter option is often cheaper, and it offers a quicker solution to deal with pressing network capacity demands while waiting to procure new infrastructure. It also prolongs the lifetime of older equipment, thus protecting investments in existing infrastructure.

2.3 Major Challenges of Bandwidth Aggregation for Concurrent Multipath Transmission

As it has been presented earlier, aggregating bandwidth from multiple network interfaces for concurrent multipath transmission can bring benefits in the form of high throughput, reliability, low-cost network capacity increase, and load balancing. However, there are some challenges that need to be addressed in order to efficiently reap the benefits of bandwidth aggregation. The challenges discussed here are packet reordering and increased battery power consumption. This research, however, aims at only tackling the packet reordering challenge. The problem of increased battery power consumption [18], albeit critical to a complete concurrent multipath solution, is outside the intended scope of the research work carried out in this thesis.

2.3.1 Packet Reordering

Packet reordering occurs when packets of the same flow somehow arrive at the receiver in the order that is different from the order in which the packets were sent at the sender [19]. That is, the sequence number of the arriving packet is lower than the sequence number of a packet of the flow, which has already arrived at the receiver. Fig. 2.2 is an illustration of packet reordering at the receiver. In the figure, the packets that have been misplaced (reordered) are shown in boldface.

Sender	**Receiver**
1 2 3 4 5 6	1 **4 5 3** 2 6

Fig. 2.2 Packet reordering illustration

Packet reordering in the context of concurrent multipath transmission is caused by simultaneous transmission of packets of the same flow across multiple network paths that have different end-to-end delays and transmission rates, thus resulting in consecutive packets of the flow arriving at the receiver out of the intended order. Packet reordering can adversely affect the performance of concurrent multipath transmission of real-time applications [19]. When packet reordering has occurred, the time taken to put the received packets in correct order increases the packets' end-to-end delays, thus causing some of the packets of delay-sensitive applications to miss their playback deadlines and ultimately get discarded. Applications that use the Transmission Control Protocol (TCP) can also be affected by packet reordering. TCP can tolerate packet reordering by a maximum of two positions, and it can be corrected by the inherent re-sequencing mechanism [19, 20]. However, packet reordering beyond two positions will be interpreted as a loss, and the transmission window can be reduced drastically. Consequently, the capacity that has been aggregated from several network paths in concurrent multipath transmission will be underutilized, and the application throughput may drop significantly. It is imperative, therefore, for any efficient bandwidth aggregation solution for concurrent multipath transmission to include effective and efficient mechanisms to minimize or eradicate (where possible) packet reordering so that the effects may not be as profound on the performance of a concurrent multipath transmission system.

2.3.2 Increased Battery Power Consumption

The battery power on mobile devices has always been a cause for concern in the design and development of high performance mobile computing solutions. During operation and idle periods, a mobile device consumes a significant amount of power, and its battery gets depleted. When the mobile device is equipped with multiple network interfaces, its battery power consumption can increase even more [18, 21]. The increase in battery power consumption inevitably reduces the mobile device's operational lifetime, thus subjecting the device's on-going communication session to premature termination, which can be quite frustrating to the users. Therefore, an efficient bandwidth aggregation solution should include mechanisms to realize concurrent multipath transmission at the minimum mobile device's battery power consumption cost possible so that the battery lifetime can be prolonged to avoid unintended multipath communication termination. The work in [18] marks an important step in optimizing mobile device's battery power consumption during concurrent multipath transmission.

2.4 Packet Reordering Metrics

Since packet reordering is the dominating challenge in the design of efficient bandwidth aggregation solutions for concurrent multipath transmission, it is important for designers to fully understand how to quantify it so that they can reliably assess the performance of their solutions and make appropriate optimizations to ensure improved performance. Several metrics have been proposed to quantify packet reordering. Such metrics include packet reordering percentage, reorder entropy, reorder density and reorder buffer density. Here, the discussion is on reorder density, reorder entropy and reorder buffer-occupancy density, which, according to [22], are deemed the most important and less ambiguous packet reordering metrics.

2.4.1 Reorder Density

Reorder Density (RD) measures the amount and extent of packet reordering in a sequence of arriving packets, i.e., the distribution of displaced packets, which is normalized to the number of packets in the sequence. An early packet has a negative displacement and a late packet has a positive displacement. RD has been demonstrated to provide comprehensive information about the amount of packet reordering. For a sequence of packets (1, 2, …, N) transmitted over a set of

network paths, the receive index (RI), matching the packet's sequence number, is assigned to help verify the arrival order. Lost and duplicate packets are not part of the RI. If the RI of packet m is $(m + d_m)$, where d_m is not zero, then packet reordering has occurred. The packet reordering event is denoted as $r(m, d_m)$. When there is no packet reordering, the sequence number of the arriving packet matches the RI. A packet is late if $d_m > 0$ and it is early if $d_m < 0$, so packet reordering is the union of the different reorder events as shown in equation (1).

$$Reordering = \cup \{r(m, d_m) \mid d_m \neq 0\} \quad (1)$$

RD is represented as a histogram of packet displacement (d_m) values, which are normalized to the total number of packets. Fig. 2.3 provides an illustration of RD calculation from the arriving packet sequence. The corresponding RI values for the sequence and the resulting displacements are shown.

Arrival Sequence	1	2	4	5	3	7	6
Receive Index	1	2	3	4	5	6	7
Displacement (dm)	0	0	-1	-1	2	-1	1

Fig. 2.3 Packer Reordering Calculation

From Fig. 2.3, the reorder set is determined and the RD is calculated as follows:

R = { (3, 2), (4, -1), (5, -1), (6, 1), (7, -1)} RD[0] = 2/7; RD[1] = 1/7; RD[2] = 1/7; RD[-1] = 3/7

For instance, packet 1 and packet 2 are not reordered, which gives RD of 2/7; packet 3 is late by two positions with density of 1/7, packet 4, packet 5 and packet 7 are early by one position each, resulting in RD of 3/7; finally, packet 6 is late by one position with the density of 1/7. As it can be observed from the given calculations, RD is not a singleton, which complicates a comparison of different multipath transmission schemes.

2.4.2 Reorder Entropy

To characterize the disorder in the arriving packet sequence using a single value for ease of comparison of multipath transmission solutions, reorder entropy (RE) has been proposed [23]. Reorder entropy is based on the RD distributions and it accounts for both the earliness and lateness of displaced packets. In fact, reorder entropy is a cumulative value, which can also determine the

breadth of packet reordering distribution. RE is calculated from RD values as follows:

$$RE = (-1)\sum_i (RD[i] \times \ln RD[i]) \quad (2)$$

In this thesis, reorder entropy is used to compare the proposed network-layer concurrent multipath transmission solution with the reference solution.

2.4.3 Reorder Buffer-occupancy Density

Reorder Buffer-occupancy Density (RBD) is another way of measuring the impact of packet reordering. RBD measures buffer occupancy frequencies normalized to the number of non-duplicate packets in the arriving packet sequence [22, 23]. RBD is especially important for predicting the amount of resources (e.g. buffer space) required to correct packet reordering. Low RBD is especially desirable for mobile devices since they often have limited storage space. The calculation of reorder buffer occupancy is illustrated in Fig. 2.4. For instance, the arrival sequence for packet 1 and packet 2 match the respective expected sequence numbers, meaning that they are in correct order and do not need to be buffered, hence the reorder buffer occupancy is 0. On the other hand, packet 4 arrives early and it has to be buffered, which sets the buffer occupancy to 1. Similarly, packet 5 is not expected, which increases the reorder buffer occupancy to 2. When packet 3—which is the expected packet—arrives, packet 4 and packet 5 are freed from the reorder buffer as they are the next expected packets. Consequently, the reorder buffer is reset to 0. The next expected packet is 6 but 7 is received, thus setting the reorder buffer occupancy to 1. When packet 6 arrives, packet 7 is freed, which leaves the reorder buffer empty.

From the reorder buffer occupancy, the reorder buffer density is calculated by normalizing the different buffer occupancies to the total number of packets as follows:

RBD [0] = 4/7 = 0.571; RBD [1] = 2/7 = 0.286; RBD [2] = 1/7 = 0.143. From these calculations, it can be observed that the reorder buffer was empty for 57.1% instances; the buffer had 1 packet for 28.6% instances and 2 packets for 14.3% instances. High reorder buffer density is indicative of high packet reordering, which is not desirable for optimal concurrent multipath transmission. For a constrained reorder buffer, high packet reordering will lead to packet loss due to buffer overflow. Therefore, multipath schemes that can avoid reorder buffer overflow must be developed.

Arrival Sequence	1	2	4	5	3	7	6
Expected Squence Number	1	2	3	3	3	6	6
Buffer Occupancy	0	0	1	2	0	1	0

Fig. 2.4 Reorder Buffer Occupancy Calculation

2.5 Bandwidth Aggregation Architecture: Functional Components

To address the challenges and reap the benefits of simultaneous use of a user's device's multiple network interfaces, efficient bandwidth aggregation architecture should be developed. Typical bandwidth aggregation architecture consists of, but not limited to the following functional elements [24, 25]: network interface selection, scheduling algorithm, packet re-sequencing unit, and network path monitor. Fig. 2.5 shows a sample block configuration of these elements.

2.5.1 Network Interface Selection

Network interface selection is responsible for choosing an optimal set of network interfaces to achieve the desired bandwidth aggregation for concurrent multipath transmission. This can be done by constructing a cost function, taking into consideration the available network interfaces and their characteristics (bandwidth, delay, loss, etc.). Then, a set that better optimizes the cost function can be selected to pool the required amount of bandwidth.

2.5.2 Scheduling algorithm

A scheduling algorithm is the core component of any bandwidth aggregation architecture. The function of a scheduling algorithm in concurrent multipath transmission is to implement packet distribution policies to decide how packets can be striped over the available network paths so that they can arrive at the receiver within their decoding deadlines and in correct sequence. For instance, packets can be striped and distributed to the available network paths based on the round robin (RR) policy, which allocates packets in order from the first network path to the last one. A variety of scheduling policies for bandwidth aggregation, albeit needing improvement, have been proposed, and they are discussed in great details in chapter 3.

2.5.3 Packet Re-sequencing Unit

Packet re-sequencing *unit* usually resides on the receiver side, and its primary purpose is to assemble arriving packets into the original packet stream, ensuring that the packets follow according to their correct sequence numbers, where lower sequence number packets must arrive before higher sequence number ones. A re-sequencing unit maintains a buffer to hold displaced packets until they can be correctly sequenced and then delivered to higher layers of the network protocol stack. Determining the size of a re-sequencing buffer is an important design issue, which can have profound impact on the performance of a concurrent multipath transmission system; a buffer that is too large can hold out-of-sequence packets for too long, thereby increasing their end-to-end delays, thus resulting in missed deadlines and spurious retransmissions for TCP applications. On the other hand, a buffer that is too small can quickly overflow, resulting in significant packet losses. Hari et al. [26] show how the size of a reorder buffer can be determined.

2.5.4 Network Path Monitoring

Network path monitoring is crucial for providing performance information about the network paths participating in the concurrent multipath transmission. The information can then be used to appropriately tune performance gains of a concurrent multipath transmission system [27]. The network path monitoring must accurately measure the network path characteristics as they change and avail the relevant measurements to other components of the bandwidth aggregation system. With the network path monitor, the selection of a proper set of interfaces and the derivation of optimal packets schedules over the network paths can be done accurately.

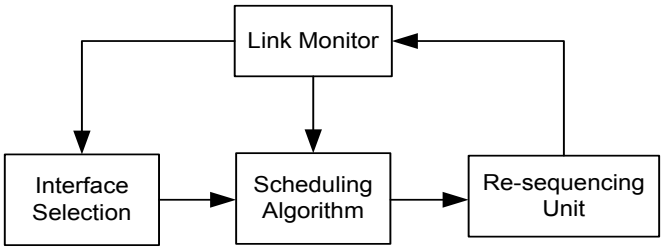

Fig. 2.5 Key Components of Bandwidth Aggregation Architecture

2.6 Chapter Summary

Bandwidth aggregation, which pools bandwidth from multiple network paths that a multi-homed user's device can connect to simultaneously, is the basis for concurrent multipath transmission, which promises performance benefits in terms of increased throughput and reduced end-to-end delays. This chapter presented an overview of bandwidth aggregation, which included a discussion of the benefits and challenges of realizing efficient bandwidth aggregation solutions in heterogeneous networks. Metrics to measure packet reordering, which is one of the most important design issues in bandwidth aggregation, were introduced. Finally, key components that characterize a typical bandwidth aggregation system have been briefly highlighted. The next chapter covers a critical review of existing bandwidth aggregation solutions for concurrent multipath transmission in heterogeneous networks.

Chapter 3

Critical Review of Existing Bandwidth Aggregation Solutions for Concurrent Multipath Transmission

For a multi-homed user's device to exploit the aggregated bandwidth capacity in a heterogeneous network, it needs to connect to several network paths simultaneously. Then, its application packets can be transmitted using one network path at a time and dynamically switching the packet transmission to the best network path any number of times during the transmission period, or the application packets can be striped across multiple network paths for concurrent multipath transmission. While this can significantly boost the operators' network capacity for improved service provisioning, it brings along challenges that need to be addressed. For instance, as mentioned in the previous chapter, when packets belonging to the same application are transmitted simultaneously via multiple network paths with heterogeneous latencies, they are likely to arrive at the receiver out of the intended order, and this can adversely affect performance of real-time and TCP applications [28]. Also, activating multiple network interfaces on a multi-homed device may significantly increase battery power consumption, thereby shortening the terminal's battery lifetime and risking premature transmission termination. To realize efficient bandwidth aggregation for concurrent multipath transmission, several bandwidth aggregation approaches have been and continue to be developed.

In the previous chapter, an overview of bandwidth aggregation and multi-homing—explaining some of the important concepts and elements to consider for concurrent multipath transmission—has been presented. This chapter provides a comprehensive and critical review of the existing bandwidth aggregation solutions in heterogeneous networks; particularly, the focus is on protocols, traffic scheduling and distribution schemes designed to exploit the aggregated bandwidth capacity. The overall goal of this chapter is to contribute to the review and analysis of the state-of-the-art research on bandwidth aggregation to enable concurrent multipath transmission for multi-homed devices. The presented critical review serves as an important reference to guide efficient future design and development of concurrent multipath transmission solutions.

The rest of the chapter is organized as follows. In section 3.1, factors considered to classify approaches to bandwidth aggregation for concurrent multipath transmission are discussed. The classification helps to better understand the characteristics, operation and implementation details of the existing bandwidth aggregation approaches. Section 3.2 provides a review of non-adaptive bandwidth aggregation schemes. Adaptive schemes are reviewed in section 3.3. Summary of the reviewed approaches is presented in section 3.4.

3.1 Classification of Bandwidth Aggregation Solutions in Heterogeneous Networks

There are numerous bandwidth aggregation schemes that have been proposed to achieve concurrent multipath transmission in heterogeneous networks. To fully grasp the operation of these bandwidth aggregation mechanisms, an appropriate classification approach is needed to relate the bandwidth aggregation schemes to a set of attributes that they have in common as a way to simplify their implementation and improvement. The proposed classification of the bandwidth aggregation solutions is based on the following criteria.

3.1.1 Adaptation to Dynamic Conditions

Bandwidth aggregation solutions, whose operation is impacted by dynamic network and traffic characteristics, are referred to as adaptive. Adaptive bandwidth aggregation approaches can achieve improved throughput and link utilization by dynamically tuning the network path selection and packet striping decisions to changing traffic and network dynamics. Adaptive bandwidth aggregation schemes are more fitting in heterogeneous wireless networks where the conditions of the wireless link are highly variable.

On the other hand, the bandwidth aggregation solutions that work based on static characteristics of the network paths are termed non-adaptive. The non-adaptive bandwidth aggregation mechanism are likely to deliver lower performance gains in heterogeneous wireless networks with highly dynamic network conditions as they do not reconfigure their resource allocation and traffic schedules. Non-adaptive bandwidth aggregation solutions may be suitable for concurrent multipath transmission in a wired heterogeneous network environment, which is more stable than the wireless environment.

3.1.2 TCP/IP Protocol Stack Layers

Both adaptive and non-adaptive bandwidth aggregation solutions can be realized at different layers of the network protocol stack such as the application, transport, and network and link layers. Implementation of a bandwidth aggregation solution at one particular layer has its own advantages and disadvantages. These are described below and summarized in Table 3.1.

3.1.2.1 Application layer Solutions

At the application layer, the application is aware of its data being striped over multiple heterogeneous network paths to optimize application performance and reliability. The application layer bandwidth aggregation techniques require the application developers to directly embed appropriate functions to handle concurrent multipath transmission in the application. This enables flexibility to fine-tune a bandwidth aggregation solution to fit the specific needs of the application. This, however, can complicate application development. It may also lead to incompatibility with existing applications. Some examples of applications that can be striped over multiple network paths for concurrent multipath transmission include XFTP [28] and GridFTP [29].

3.1.2.2 Transport layer Solutions

Most transport layer bandwidth aggregation approaches focus on TCP, where application data can be split into multiple TCP sessions, which can be delivered over multiple heterogeneous network paths. TCP bandwidth aggregation provides reliable concurrent multipath transmission at the transport layer. However, it may compromise interoperability between existing TCP based network settings. Since TCP is mostly not suited for timely delivery of data, time-sensitive interactive applications may not benefit from TCP-based concurrent multipath transmission.

There are also some transport layer solutions that are based on stream control transmission protocol (SCTP) [30]. SCTP enables multi-streaming and multi-homing to facilitate the establishment of several transport layer sessions allowing for segments of the same application to be sent concurrently over multiple network paths. The performance of bandwidth aggregation based on TCP and SCTP may be affected adversely by reordering. Therefore, efficient mechanisms to reduce reordering are necessary for transport layer applications to realize the benefits of concurrent multipath transmission. Besides the traditional TCP and SCTP, the IEFT Multipath TCP (MPTCP) [31] has emerged and it promises more efficient bandwidth aggregation.

3.1.2.3 Network layer Solutions

Network layer based bandwidth aggregation provides packet level traffic striping over multiple network paths to improve performance. Bandwidth aggregation at the network layer is flexible and it can apply across different network domains, infrastructure and protocols. However, network layer bandwidth aggregation is not immune to reordering, which can degrade performance. To mitigate packet reordering, additional packet processing, i.e., marking packets with sequence numbers to simplify re-sequencing at the receiver before packets can be passed to higher protocol layers. Moreover, a multipath packet scheduler can be used to effectively combat packet reordering.

3.1.2.4 Data link layer Solutions

Link layer bandwidth aggregation solutions aggregate multiple data link channels into a single logical link that is large enough to handle high capacity demands. Here, the link layer protocol data units (PDUs) belonging to a single IP flow can be split across multiple channels for improved performance. Some notable solutions for link layer bandwidth aggregation include Multilink PPP [32]. There is also the idea of a generic link layer (GLL) [33]. The GLL aims at providing all-encompassing link layer processing functions to stripe frames over multiple heterogeneous radio interfaces to leverage various forms of multi-radio diversity for improved performance. The link layer bandwidth aggregation approaches can, however, be limited to local domains due to the need for special hardware or software.

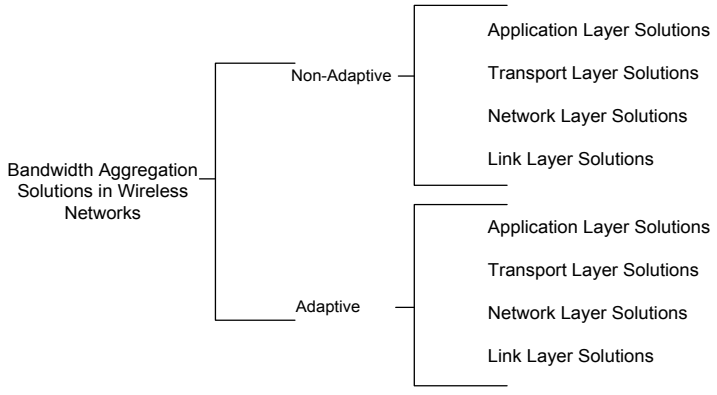

Fig. 3.1 Taxonomy of Bandwidth Aggregation Solutions

Table 3.1 Bandwidth aggregation at different layers of the network protocol stack

Layer	Description	Advantages	Disadvantages
Application	Application is aware of multiple interfaces, and can split the traffic into several application layer protocol data units, which can be transmitted simultaneously via the interfaces.	Finer granularity and efficient application specific optimizations due to full knowledge of application characteristics	Increased application complexity; Compromised interoperability with existing applications; head-of-line blocking at the transport layer
Transport	Multiple transport layer connections are created to transmit application traffic.	Reliable multipath transmission in the case of TCP and SCTP	Compromised interoperability with existing TCP based infrastructure
Network	IP packets from the same transport layer session are transmitted across multiple network interfaces.	Transparent to higher layers; compatible with existing infrastructure	Poor TCP performance due to high packet reordering
Link	Multiple links are bundled into a single logical communication link.	Higher utilization of the aggregated capacity	Limited to tight-coupled networks belonging to the same operator

The comprehensive and critical review of the different bandwidth aggregation schemes in the next section follow the proposed classification criteria to characterize existing bandwidth aggregation schemes in heterogeneous networks. Firstly, non-adaptive bandwidth aggregation solutions are discussed at the different layers of the network protocol stack. Secondly, adaptive schemes are discussed in great details. Fig. 3.1 illustrates the proposed taxonomy of the reviewed bandwidth aggregation approaches.

3.2 Non-Adaptive Bandwidth Aggregation Solutions

Non-adaptive bandwidth aggregation solutions perform concurrent multipath transmission over multiple heterogeneous network paths based on static network and traffic characteristics. It should be noted that non-adaptive bandwidth aggregation might offer sub-optimal performance for a highly variable network environment such as the wireless network setting. The different non-adaptive bandwidth aggregation solutions at different layers of the network protocol stack are discussed as follows.

Application Layer: To accomplish bandwidth aggregation at the application layer, a striping mechanism is included in the application development to allow for the application data to be segmented into several application flows, which can be transmitted across multiple network paths simultaneously. Early developments of application layer based bandwidth aggregation attempted to stripe application data across multiple logical channels belonging to the same network interface. Such developments include XFTP [28] and GridFTP [29] to enable concurrent multipath transfer of FTP data. Parallel Sockets (PSockets) [34] solution was also developed to create multiple parallel sockets for data transfer across multiple logical connections. The drawback of these proposals is relying on bandwidth from a single network interface, thus failing to leverage multiple physical network interfaces modern communication devices.

Among the first application layer based bandwidth aggregation solutions to use multiple physical network interfaces is MuniSocket [35]. MuniSocket was developed as a middleware approach that facilitates big data transfers across multiple network paths. MuniSocket maintains a pair of threads to send and receive simultaneously over multiple network interfaces. Moreover, a counter is used to sequence message fragments to send over multiple paths. When the message fragments get to the receiver, the receiving thread retrieves the segments from the different interfaces and keeps them in the receiver buffer until they are reordered correctly. MuniSocket also includes reliable multipath transmission through acknowledgment triggered retransmissions. To ensure proper load distribution across the multiple network paths, MuniSocket allows only the sending threads that are linked to the least loaded networks to accept message fragments for transmission. Elegant as MuniSocket may be, it lacks the ability adapt to changing network conditions. Consequently, MuniSocket is likely to achieve sub-optimal performance in highly variable setting such as the wireless environment.

Transport Layer: The IETF has published specifications for a new transport layer protocol to exploit multiple network interfaces on modern devices to improve through and robustness [36]. The protocol is called Multipath TCP (MPTCP), which is essentially the improvement of the traditional TCP. Fig. 3.2 illustrates the components of MPTCP. The rationale of the IETF MPTCP is to segment TCP application data into several sub-flows, which can be transmitted simultaneously across multiple network paths. To establish a multipath session between the sender and the receiver, the MPTCP receiver transmits a SYN segment to the receiver. If the receiver supports MPTCP, it replies with a SYN that indicates MPTCP capability. Initially, communication between the source and destination occurs via a single interface. More interfaces can be added using a SYN segment with MP_JOIN field. MPTCP performs sequence numbering at two levels: MPTCP level to facilitate in-order segment delivery, and sub-flow level to administer retransmissions within individual TCP sub-flows.

In its current state, the MPTCP only employs basic interface selection and traffic scheduling mechanisms to transmit across multiple paths with heterogeneous path characteristics. Some of the first implementations of the IETF MPTCP has been reported in [37] with performance analysis presented in [38]. It has been demonstrated that MPTCP can leverage multiple paths to improve throughput and reliability for TCP applications. However, the performance of MPTCP degrades significantly when the network paths have considerably diverse path characteristics.

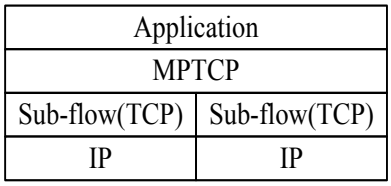

Fig. 3.2 MPTCP Architecture [37]

Network Layer: The most natural and notable multipath scheduling policy is the round robin (RR). The round robin, however, is non-adaptive. The RR multipath scheduler stripes packets of the same application across multiple network paths in equal portions and in cycles. The round robin assumes the network paths possess homogeneous link characteristics. It also assumes equal sized packets. Fig. 3.3 illustrates the operation of the round robin scheduling mechanism over three network paths. In the first scheduling round, packets 1 through 3 are striped in order and equally across the paths. The same operation is performed in round 2 for Packets 4 through 6.

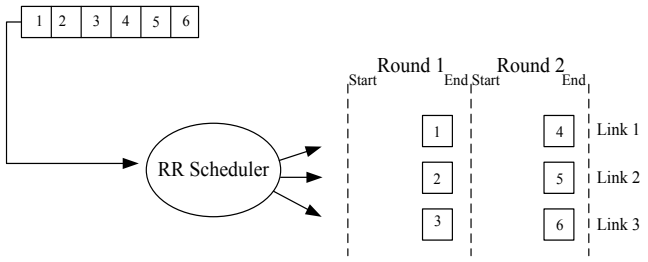

Fig. 3.3 The operation of RR at the sender

The RR scheduler is attractive due to its simplicity as it stripes packets across multiple paths incurring per packet computational complexity of O(1). However, the performance of multipath round robin in terms of packet reordering, throughput and load balancing can degrade significantly in the face of variable packet sizes and dynamic path characteristics. When a low-bandwidth network path is assigned larger load to transmit than high-bandwidth paths, the traffic load cannot be balanced efficiently. Furthermore, packets on the low-bandwidth path may suffer increased delays. The delay disparity between the high-bandwidth and low-bandwidth paths often leads to increased packet reordering. Increased packet reordering requires a large reorder buffer at the receiver, which may not be desirable for memory-constrained devices. A large reorder buffer can worsen end-to-end delays since it allows for reordered packets to be stored for an extended period of time.

To support concurrent multipath packet transmission over network paths with heterogeneous transmission rates, there are variants of RR, such as weighted round robin (WRR) [39] and surplus round robin (SRR) [26], which can be employed. Concurrent multipath transmission based WRR striping assigns packets from a flow to multiple paths using a normalized weight. The path with a larger weight carries more packet load to the destination in a scheduling round. A path's weight can be defined as a function of the path's offered bandwidth. The WRR's weighted packet striping can accomplish better load balancing than the simple RR. However, variable length packets can degrade the WRR's load balancing and high packet reordering performance. A variant of RR that can handle variable length packets is SRR. The primary purpose of SRR is to achieve concurrent multipath transmission that efficiently balances traffic load across multiple paths. For packet reordering, SRR relies on the user of reorder buffer at the receiver. This may not be desirable for handheld devices with limited resources.

Besides the round robin and its variants, other non-adaptive multipath packet scheduling policies have been proposed. For instance, Kaspar et al. [40] proposed a non-adaptive packet scheduler to address packet reordering in order to efficiently benefit from concurrent multipath transmission. The proposed multipath scheduling mechanism is based on a once-off estimation of end-to-end delays on the selected paths. The proposed multipath scheduler attempts to reduce packet reordering by delaying packets on the faster paths to ensure that they incur similar delays to packets on the slower network paths. Kaspar's multipath packet scheduler achieves lower packet reordering delay than the traditional round robin. However, like other non-adaptive scheduling mechanisms, the proposed scheduling strategy is likely to experience reduced throughput and increased packet reordering when path conditions change over time.

Another non-adaptive multipath packet scheduling framework is presented by Kim et al. [41]. The framework consists of two functional components: bandwidth estimation and packet partition scheduling. The bandwidth estimation unit calculates the amount of traffic load in bytes that can be assigned to a path without causing congestion. The partition packet scheduler also attempts to distribute the packets across the different paths in the manner that efficiently balances the load. To achieve load balancing, the partition scheduler maintains a counter (a weight associated with a path based on available bandwidth) to check if a network path can accept any more bytes of the packet. If the current path has a lower counter and cannot accept the incoming packet, the next path with a larger counter will be selected to transmit the packet. Although the proposed scheduling mechanism can accomplish more efficient load balancing than simple weighted multipath packet schedulers, such as WRR, its performance may degrade as a result of high packet reordering in rapidly changing channel dynamics as in a wireless environment.

Link Layer: Koudouridis et al. [42] introduce the idea of generic link layer (GLL) to enable multi-radio cooperation at the radio level. Among other features, the GLL enables multi-radio transmission diversity (MRTD), which is the sequential or parallel use of the aggregated bandwidth capacity to transmit a traffic flow. The important GLL functions for realizing different forms of MRTD include: access selection to schedule users' data on appropriate RATs; performance and monitoring to provide performance information to the accession selection unit so that transmission schedules that match network and traffic dynamics can be drawn; flow and error control to regulate data flow and ensure error-free transmission on the links.

Even though the GLL concept is an innovative step towards enabling efficient bandwidth aggregation in heterogeneous wireless networks, challenges in the form of packet reordering and unbalanced load distribution are still a concern. This is especially true for naïve parallel MRTD that blindly disperses packets across multiple links [43]. Yaver and Koudouridis [43] show that parallel MRTD without adapting to varying channel and traffic conditions can only achieve average delay and drop probability, which are, respective averages of average delay and drop probability of individual links. Moreover, naïve parallel MRTD does not show any quantitative gains in throughput. Therefore, to enhance the performance of parallel MRTD, intelligent and dynamic traffic distribution mechanisms are needed.

3.3 Adaptive Bandwidth Aggregation Solutions

Adaptive bandwidth aggregation solutions deal with the short-comings of non-adaptive schemes and consider changing traffic and path characteristics to determine optimal resources allocation and scheduling decisions. Similar to non-adaptive, adaptive bandwidth aggregation schemes can be designed and implemented at different layers of the network protocol stack.

Application Layer: Luo et al. [44] presented one of the first application layer based bandwidth aggregation approaches to stripe application data across tight-coupled wireless LAN (WLAN) and 3G heterogeneous network. The proposed mechanism employs a joint session scheduler (JOSCH) to segment application data into important and optional sessions. For example, a video session can be split into base and enhancement layer data; and Hyper Text Transfer Protocol (HTTP) session can be segmented into main and inline data objects. The important data streams are transmitted via the reliable UMTS domain while the optional information is served on the WLAN link. Sending data traffic from the same session simultaneously via heterogeneous network paths can lead to loss of synchronization, thereby affecting the play-out performance of the application. To circumvent this, Luo et al. devise a periodic interactive method, which uses a small data unit, referred to as a training sequence, to estimate average latency on each network link. The average delay estimate is then employed to determine traffic quantum sizes for the aggregated network paths in a manner that minimizes delay difference. Luo et al. also adopt a course-grained flow-level session splitting, which may lead to inefficient load balancing when the flows have different sizes. Furthermore, the training sequence employed to determine delay information is too small to produce accurate delay estimates.

Ghareeb *et al.* [45] proposed concurrent multipath streaming of SVC-encoded video in multipath overlay networks. The proposal uses available bandwidth and the multi-homed device's capabilities to determine the number of video layers to transmit over multiple paths. During the multipath streaming process, the video load can be adjusted to match the varying bandwidth. Furthermore, the multipath streaming system optimizes utilization of bandwidth by mapping a video layer to a path offering the smallest available bandwidth, which can meet the video layer's play-out requirements. However, the proposed solution does not consider network path loss rates when striping the SVC video layers across multiple paths. That can risk the loss of the important parts of the video data, as the data may be sent over the paths experiencing high loss. When the important part of the video data gets lost, the resulting drifting errors can cause severe video distortion, thus producing poor video quality. Also, the proposed multipath video streaming approach only employs flow-level striping of the video data over the network paths, which may result in poorly balanced traffic load.

In [46], concurrent SVC transmission framework, which chiefly employs cross-layer control information and a simple video striping function, has been presented. The cross-layer signalling is used to collect context information from the multi-homed device in order to determine the status of the device's network interfaces. When several network interfaces are in use, the striping function randomly schedules the base layer and the enhancement layers over the different network interfaces for concurrent multipath transmission. During the concurrent multipath transmission, if some of the interfaces become inactive leaving only a single interface active, the proposed streaming approach attempts to transmit the base layer only. The proposed streaming solution does not consider any specific path characteristics, such as bandwidth and loss rates, in its operation; therefore, it may not result in optimal performance. Also, the proposed scheme may not efficiently utilize the multiple network interfaces on a multi-homed device since it only uses flow-level splitting to distribute the video layers across the network paths.

Nightingale *et al*. [47] presented a concurrent multipath transmission solution to stream SVC video in multi-homed network mobility (NEMO) setting, where the SVC stream is sent over one path at a time, switching to a different path when path conditions vary unfavorably to the current network path. However, the solution does not fully utilize the aggregated bandwidth, as it depends only on one high bandwidth path at time to send the video bit-stream and leaves the other network paths idle. It also does not consider loss rates, thus making it undesirable in lossy wireless

networks. Although the solution uses path switching cost to correctly determine packet delivery time, it does not remove the cost. Considerable switching overhead can affect performance of the concurrent multipath streaming process. In [48], path switching overhead is eliminated by splitting the video application into sub-flows and pinning each sub-flow to a specific network for the duration of the video transmission. A signalling solution has been presented in [49] to provide performance information that can be used to switch video layers to higher capacity interfaces for improved performance.

Xue et al. [48] proposed a bandwidth aggregation solution that uses distributed re-enforcement learning to stripe traffic load over multiple network paths. The proposed scheme aims at providing QoS guarantees as well as efficient load balancing. The presented solution splits scalable data formats, such as H.264 video, into sub-layers that can be sent over multiple paths and reassembled at the destination. To achieve this, each terminal is associated with an agent to determine a suitable traffic scheduling strategy. The agents learn and exchange the learned information with each other. The learned information is abstracted into Q-values stored in the Q-repository on the network. The traffic scheduling problem is viewed as a collection of states, actions, and rewards. The states depict session arrivals and departures; actions are basically traffic scheduling and striping strategies applied on arriving sessions; rewards represent user experience (UE) resulting from a certain set of striping strategies. The job of the agents is to discern the current state of the network and then adaptively stripe sessions over available network paths to enhance UE. The proposed re-enforcement learning solution can offer higher quality of experience than random traffic scheduling mechanisms. However, the coarse-grained flow-level load balancing may yield sub-optimal performance when flows have varying sizes.

Qureshi and Guttag [49] developed a middleware solution named Horde. Horde allows the application to control how the data can be split across multiple network paths simultaneously. Horde employs Network Channel Manager functions to monitor performance characteristics of the available network connections. The functions also handle congestion control. The network path characteristics from the channel managers can be applied by the scheduler to derive efficient scheduling decisions, thus ensuring that application data units are well balanced over the network paths and they can arrive at the destination in sequence. Most importantly, the scheduling decisions are influenced by the application's performance objectives. The objectives are depicted using a specification language as illustrated in Fig. 3.4.

```
Objective    {
    context  {
        adu: foo { (stream_id == 17) &&
                   (frame_type == "I")  }
        adu: bar { (stream_id == 17) &&
                   (frame_type != "I")  }
    }
    goal  { prob (foo::lost?)
          < prob(bar::lost?) }
    utility  { foo { 100 } }
}
```

Fig. 3.4 Application's objective specification [40]

The objective specification in Fig. 3.4 informs the scheduling engine that the application scheduled is a video data of type I from stream 17; the data should have loss probabilities smaller than other data types. Using the specification, the scheduling mechanism can determine striping strategy that can better optimize the application's objective. Applications that do not have specific objectives are striped across the paths in a round robin manner. Although Horde is elegant and efficient, it makes application development more complex. Furthermore, current applications would have to be redesigned to use Horde's multipath functionality.

Kaspar et al. [50] analysed the challenges and benefits of concurrent multipath transmission to optimize video-on-demand playback. The analysis uses an adaptive application layer multipath solution that can enable progressive download of multimedia data over multiple network paths to the receiver. The authors pinpoint long startup latency and large buffer requirements as the main challenges of continuously downloading multimedia data over multiple paths. Startup latency refers to the waiting time before the downloaded video can be played back. The wireless channel variations make it challenging to predict optimal startup latency. However, the optimal playback bitrate, which is bounded by throughput of the aggregated bandwidth, can be applied to determine the desirable startup latency. By configuring high playback bitrate (but below the aggregated bandwidth capacity), startup latency can effectively be minimized, and this is important for short video clips, where the user anticipates immediate startup.

To keep the received data (usually out-of-order data) that cannot be played out yet, a receiver buffer is used. The buffer size is reported to increase with segment size and the number

of used network interfaces. A buffer that is too large is not desirable for handheld devices, which have limited resources, including finite battery power. There is also a correlation between segment size and startup latency. The important task is how to find an optimal segment size that can result in optimal startup latency and buffer size. While the required buffer size can be affected by the number of interfaces used, startup latency is affected more by the difference in delay between the interfaces. For instance, large delay difference can lead to long startup latency, which can negatively affect the perceived video quality. Therefore, finding the right number and combination of network interfaces is also an important design factor for achieving efficient bandwidth aggregation.

Based on the analysis results in [50], a pipelining solution [51] is developed to meet the requirements of progressive video download and playback over multiple network interfaces. The designed mechanism enable the client to request for the next byte range while downloading the current video segment, thus reducing the server's idle time and increasing the aggregated throughput for small segment sizes. Although the solution can enhance throughput when using small segment sizes, there is a risk of selecting too small segment size, thereby resulting in too fast processing by the server and not allowing adequate time for the client to pipeline the next request. To resolve this issue, the amount of pipelined data must be greater than the path's bandwidth-delay product. The presented solution accomplishes this by pipelining as many segments as possible. It is noted that using fixed segment size may result in inefficient adaptation to path heterogeneity.

Transport Layer: Casetti and Gaiotto [52] presented enhancements to SCTP to use multiple paths for concurrent multipath transmission. They call to the enhanced SCTP Westwood SCTP (W-SCTP). The goal of W-SCTP is to ensure balanced traffic across several paths in concurrent multipath transmission. W-SCTP uses a bandwidth-aware scheduling policy, which is deployed at the destination. W-SCTP uses multiple transmit buffers, and each session within the W-SCTP association is handled independently. The SCTP selective acknowledgements (SACKs) management is also improved to work across multiple transmit buffers. To send a data chunk, W-SCTP determines the chunk's delivery time across all the available network interfaces, and the interface with the shortest delivery time is chosen to transmit the chunk. The process is repeated until the congestion windows of the different paths have been filled. At the destination, a common association buffer is used to keep the data chunks received from multiple network interfaces. Although W-SCTP is an important improvement for SCTP to support of concurrent multipath

transmission, it does not include explicit detail of how to resolve potential segment reordering. Furthermore, it does not consider all the critical path characteristics, such as loss rates, when deriving data chunk striping decisions. Therefore, it may not accomplish optimal concurrent multipath transmission.

In [53], Casetti and Gaiotto improved their work in [52] to achieve concurrent multipath transmission using SCTP with Partial Reliability. The new solution is referred to as Westwood SCTP-PR. The main purpose of Westwood SCTP-PR is to provide support for real-time applications by leveraging SCTP's Partial Reliability extension. To deal with long delays, which may affect the concurrent multipath transmission of real-time multimedia applications, they propose to remove SCTP retransmissions. Westwood SCTP-PR periodically builds a list of available network paths and frequently relegates stale paths from the list to assure robustness against undesirable network conditions.

A load sharing SCTP (LS-SCTP) is presented in [54] to enable end-to-end data transport over multiple paths. The main functional elements of LS-SCTP are: flow and congestion control; path assignment and monitoring. LS-SCTP separates congestion and flow control by performing flow control on association basis, whereas congestion control is done on path basis; that is, the sender maintains a separate congestion control window for each link, thus providing the sender with a logical congestion window whose size is an aggregate of the individual congestion windows. To support load balancing across multiple paths, LS-SCTP defines a 'load sharing chunk', which is associated with a path by path identity (PID). PID identifies the path that is used to transmit the data chunk. Within the same path, data chunks are identified by path specific identifiers called path sequence numbers (PSNs). The path assignment module then allocates appropriate paths to the data chunks, assigning suitable PID and PSN for each chunk. Path assignment is done based on available bandwidth on the paths.

Instead of distributing data chunks over the paths in a round robin manner, which may limit multipath transmission throughput to the slowest path, LS-SCTP allocates chunks to the paths based on the current congestion window of a path; specifically, the ratio of congestion window to the round trip time is used. To mitigate the reordering that may occur as a result of different RTTs across the paths, LS-SCTP uses the association buffer at the receiver, whose size increases with the amount of reordering. However, a large buffer can increase end-to-end delays, thus affecting

the efficiency of multipath communication. LS_SCTP adapts to varying path characteristics by frequently monitoring the paths to gather channel information and then adjust the allocation ratios accordingly. A similar bandwidth aggregation approach to LS-SCTP is concurrent multi-path SCTP (cmp-SCTP) in [55]. The glaring difference between the approaches is cmp-SCTP's use of multiple send buffers, where a send buffer is maintained for each path. The benefit of using separate send buffers is that when head-of-line (HOL) blocking occurs, it is pinned to a specific connection on a path and does not affect connections on other paths.

Hasegawa et al. [56] introduced multipath TCP communication scheme called Arrival-Time matching Load-Balancing (ATLB) to deliver data segments to the receiver in correct order. ATLB assigns a score to each path based on estimated end-to-end delay; the lowest score corresponds to the lowest delay. Based on this, ATLB schedules a data segment on the path that has the lowest score. ATLB includes a reorder buffer at the receiver to deal with residual segment reordering that may occur. ATLB handles path failures by maintaining a timer that expires after a carefully set timeout period. If a segment, just after the timeout, is not received properly, the path is assumed to have failed. To recover from failure, ATLB probes the failed path periodically, calculating packet loss rate. If the loss rate is small enough, data transmission can resume on the path; otherwise, the path is removed the list of available paths. Despite its robustness against failure, ATLB may not accomplish optimal performance because the path scores are based only on end-to-end delays, omitting other important channel characteristics (ATLB uses loss rates only for recovering from failure, not for deciding the best path for a segment).

Hsieh and Sivakumar [57] presented a new transport layer protocol called parallel TCP (pTCP) to enable applications to exploit the mobile terminal's multiple interfaces for performance improvement. pTCP is implemented as a wrapper around a modified TCP called TCP-virtual (TCP-v). For every socket that an application opens, pTCP creates and maintains a TCP-v connection on each interface that is used to aggregate bandwidth. pTCP controls what data should be sent through the available interfaces, whereas TCP-v determines the volume of data that can be sent through the interfaces. Thus, reliability and congestion control can effectively be separated. pTCP stripes application data over active TCP-v connections based on available space in their congestion windows. That is, data can only be allocated to a TCP-v connection if there is enough space in the congestion window. The adopted congestion window based splitting assumes that the congestion window is a true reflection of the bandwidth-delay product, which may not be the case

at times. Consequently, an overestimation of the congestion window may drive the TCP-v link into congestion, resulting in undesirable delay and loss. When undesirable loss and delay occur on a path, pTCP reassigns the data on the affected TCP-v to another TCP-v with enough free space in the congestion window. However, if a path shuts down completely, data in the congestion window will stall. To deal with this, redundant striping of the data is performed. Even though pTCP may be efficient in handling congestion across the paths, it does not incorporate any mechanism to assure in-order data delivery to the receiver.

In [58], another new TCP-based multipath transport protocol—*concurrent multipath real-time TCP (cmpRTCP)*—is proposed to transmit real-time data streams via multiple paths. cmpRTCP uses congestion window manager to dynamically monitor congestion status of the available paths. The information from the congestion window manager is used by a real-time scheduler to efficiently distribute packets over the paths. The proposed real-time scheduler schedules a burst of packets across the paths in a round robin manner, filling each path to full capacity before moving onto the next one. This can, however, lead to unbalanced load and poor utilization of the paths; for instance, when the burst can fit in one path, the other paths will be idle. cmpRTCP reacts to missing packets by gathering the paths with missing packets, ordering them in the increasing number of missing packets, and filling them up in the same order. This can reduce the probability of loss in the next transmission rounds. cmpRTCP does not include any mechanism to ensure in-order delivery at the sender; instead, it relies on the receiver buffer to correct possible reordering. For every packet that is transmitted to the receiver, a real-time delay tolerance limit (RTDTL) is set, and the packets arriving beyond their limit are discarded from the buffer immediately. The packets that arrive within their RTDTL and in correct order are delivered to higher layers; otherwise, they are kept in the buffer until reordering can be corrected, or until their RTDTL expires. This ensures that the reorder buffer space is almost constant regardless of the amount of reordering.

Farhah et al. [59] proposed a scheduling algorithm, Forward Prediction Scheduling (FPS), to transmit multiple SCTP sessions over multiple links, ensuring that the segments traversing different paths arrive at the receiver in correct order. FPS consists of a scheduling module to dynamically estimate end-to-end latencies of the selected paths and then transmit the segments over the paths in a manner that can minimize reordering. The main idea behind FPS is to determine the amount of data that can be sent on the fast path before the arrival of data on the slow path. For

instance, consider two paths *i* and *j*, with latencies RTT_i and RTT_j respectively, where $RTT_i < RTT_j$. The latency of a path is a function of the path's round trip time (RTT). To ensure in-order delivery, the segments to be sent on *j* are advanced by the number of segments delivered on *i* before reception on *j*. The path latencies are updated according to equation (1).

$$RTTi = \alpha RTTi + (1-\alpha)RTTi \quad (1)$$

where α is a constant between 0 and 1, which determines the rate of adaptation of the latency estimation.

The proposed FPS is compared in simulation to W-SCTP, and it achieves higher throughput. It also achieves lower reordering delay than multipath round robin SCTP [60]. However, the FPS's adaption to varying path conditions is limited to path latency (computed as a function of RTT), disregarding other important path characteristics, such as loss rates. As a result, FPS may not achieve optimal multipath transmission. A more adaptive version of FPS is presented in [61]. Here, a cross-layer mechanism is proposed to enable FPS to use link-layer information to better adapt to varying wireless channels. Thus, robust multipath transmission can be accomplished.

Wang et al. [25] presented a generic transport layer model to enable bandwidth aggregation over multiple heterogeneous wireless networks. The proposed generic transport layer model consists of two functional elements: multiple transmission control protocol (MTCP) and common transport control (CTP). MTCP controls simultaneous connections over multiple radio access networks (RANs). That is, it schedules data on multiple RANs; it also establishes and manages multiple transport layer connections over the RANs. CTP is responsible for path specific connection management.

The performance of the proposed transport layer model depends mainly on a scheduler, which is implemented as part of the MTCP. The objective of the scheduler is to minimize delay difference between the selected RANs, thus reducing segment reordering at the receiver. This is achieved by dynamically allocating the traffic load to the RANs according to appropriate allocation ratios. The allocation ratios are calculated using estimated transmission rates and delays. To adapt to the varying network conditions, segment based feedback approach is used. This divides traffic into fixed length segments, which are transmitted via the RANs. On receiving a segment, the receiver sends back transmission delay information to the sender to adjust the allocation ratios.

The scheduler transmits a segment by diving it into sub-segments, which can then be delivered simultaneously over multiple RANs. The sub-segments are numbered and time-stamped for proper reception by the receiver. The transmission process is illustrated in Fig. 3.5. MTCP may not be robust against losses, as it does not consider transmission and congestion losses when computing traffic allocation ratios.

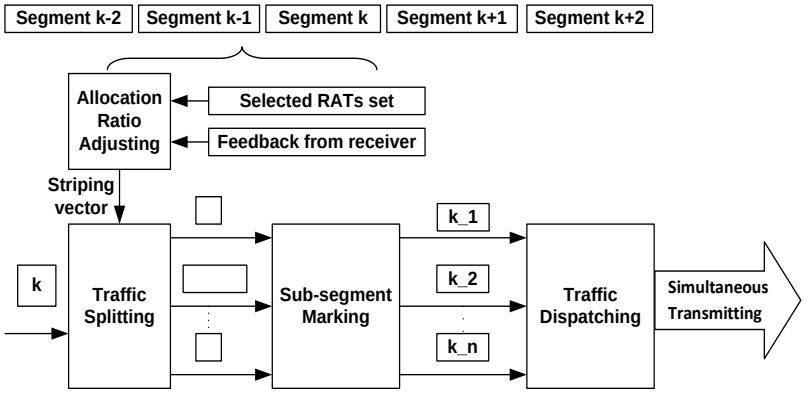

Fig. 3.5 Segment Transmission Process [25]

Network Layer: The most notable network layer-based solution to enable efficient and adaptive packet transmission using aggregated bandwidth capacity is the earliest delivery path first (EDPF) scheduling proposed by Chebrolu and Rao [24, 62]. The objective of EDPF is to deliver packets over multiple paths within the shortest time possible while ensuring that the packets arrive at the receiver in correct order. EDPF's rationale is to dynamically estimate the delivery time of the next packet on each link. Then, EDPF transmits the packet through the path that delivers it the earliest. The delivery time is estimated according to equation (2).

$$d_i^l = MAX(a_i + D_l, A_l) + \frac{L_i}{B_l}, \qquad (2)$$

where d_i^l, a_i, and L_i denote the estimated delivery time of packet i via path l, the arrival time of packet i at the network proxy, and the length of packet i respectively. D_l, A_l, and B_l represent the delay from the proxy to the base station on path l, the time when path l is available for transmission, and the bandwidth of path l respectively. The first component of the equation determines the time at which transmission can resume on path l, while the second one returns the packet transmission time along the path. EDPF, therefore, schedules a packet on path p, satisfying

$$p = \{l: d_i^l < d_i^m, l \leq m \leq N\}, \quad (3)$$

where N is the number of interfaces and p is the path with the earliest delivery time. d_i^l is computed dynamically before the next scheduling round, ensuring that all the available paths can be used while preserving the order in which packets are delivered to the receiver. The work complexity of EDPF is O(n) since the packet's delivery time is computed for each of the available links to determine one with the shortest time.

EDPF outperforms simpler schedulers, such as static RR-based multipath schedulers in terms of packet reordering and end-to-end delay. However, it does not fully utilize bandwidth of the selected set of paths, as it uses only one interface at time to transmit packets, switching to the best interface any number of times during the transmission session. In addition, EDPF considers only path bandwidth and latency, leaving other important link characteristics, such as packet loss rate. Therefore, it may achieve sub-optimal performance gains in the presence of high losses.

In [63], Chebrolu and Rao presented a mechanism to deliver interactive video traffic over multiple paths to enhance overall video quality. The proposed scheme includes a frame discard strategy (Min Cost Drop) to selectively discard frames, ensuring that high priority frames can be delivered to arrive within their playback times. Min Cost Drop (MC-Drop) exploits the high correlation of frame sizes across group of pictures (GOPs) to predict the sizes of future frames. Based on this, a decision on whether or not to drop a packet can be made. MC-Drop discards the current packet if, by doing so, the future high priority frame can meet its playback time. Moreover, MC-Drop drops all packets of the frames, together with their dependents, that cannot meet their deadlines. Packets of the frames that can meet their deadlines are scheduled over multiple paths according to EDPF. Even though the scheme uses application layer information to drop packets, its striping point is at the network layer. The presented selective frame discard approach achieves higher video quality in terms of peak signal-to-noise-ratio (PSNR) than most techniques that attempt to transmit all the application frames. However, the scheme experiences an increase in frame loss as the number of parallel paths with asymmetric characteristics increases. Therefore, it is important to determine the optimal number of paths to use to maintain acceptable video quality.

In [64], Liu et al. presented an improvement of EDPF, where the proposed scheduler considers transmission rates and losses to estimate packet delivery time. Packet delivery time can be determined as

$$d_i^l = D_l + d_l + \frac{L_i}{R_l(1-\alpha_l)} \quad (4)$$

where D_l is the one way wire line delay in the core network of path l, d_l is the one way wireless delay on the path, R_l is the data rate of the wireless interface l, α_l is the packet loss rate on path l, and L_i is the length of packet i. Similar to EDPF, Liu's scheduler selects a packet for transmission according to (3), where d_i^l is estimated using (4). The two schemes achieve similar throughput gain at low loss rates; but Liu et al.'s scheme is superior to EDPF during high loss rates. However, Liu's scheme only considers losses due to wireless transmission errors, ignoring losses caused by congestion. Therefore, its optimality may diminish considerably during high congestion periods.

Taleb et al. [65] and Fernandez et al. [66] considered time slot allocation of the aggregated bandwidth capacity and propose a scheduling mechanism called time-slotted earliest deadline path first (TS-EDPF) to exploit the aggregated capacity. In a time-slotted system, each terminal is assigned a time slot to access the wireless channel. To make accurate estimate of the delivery time for the next packet on the channel, the network proxy needs to know the start and end of the time slot that is assigned to the terminal. The delivery time can then be estimated according to equation (5) below.

$$d_i^l = g(f(MAX(a_i + D_l, A_l), l) + \frac{L_i}{B_l}, l) \quad (5)$$

where $f(\ldots)$ is a function that returns the next valid time at which transmission can begin at the base station on path l. $g(\ldots)$ returns a valid time at which the transmission of packet i can complete. The functions are defined to ensure that packet transmission occurs within the assigned time slot. TS-EDPF achieves lower reordering delay and packet losses than EDPF in a time-slotted wireless network system. However, like EDPF, it is not optimized to handle cases where congestion and wireless path losses may be high.

Another improvement of EDPF—Packet-Pair EDPF for TCP applications (PET)—is studied in [67]. PET estimates the delivery time of packets by sending packet pairs to the receiver, which computes inter-arrival time between the pairs and reports to the network proxy using signaling acknowledgements (SIG-ACKs). The proxy uses the inter-arrival information to estimate bandwidth and delay on the paths. Then, EDPF is used to schedule packets for transmission over the available links. The residual delay at the receiver is hidden from TCP using a buffer management policy (BMP), which holds out-of-order packets and reorder them correctly

before delivering them to TCP. To detect and handle lost packets, the proposed BMP uses timer-based and comparison-based loss detection techniques. The *timer-based* loss detection associates a sequence number (*N*) with a timer. When the timer expires and *N* has not been received, it is assumed that the packet was lost. Then, buffered packets can be delivered to TCP so that duplicate ACKs can be sent to trigger fast-retransmit. *Comparison-based* approach, on the other hand, assumes that packets always arrive in correct order; i.e., when sequence numbers greater than *N* arrive at the receiver, it is assumed that *N* was lost. To avoid burstiness of TCP ACKs to the sender or packets from the reorder buffer to higher layers, a technique similar to ACK pacing [68] is implemented. Even though PET, in combination with BMP, can effectively handle TCP traffic, it has similar drawbacks to EDPF because its operation assumes lossless channels, which is not practical especially in wireless networks.

Sumet et al. [69] proposed a load distribution model called effective delay-controlled load distribution (E-DCLD) for multipath packet transmission. The objective of the model is to minimize latency difference among the paths in order to reduce packet reordering at the receiver and to efficiently balance load across the paths. E-DCLD consists of three functional components: traffic splitter to derive allocation ratios for the paths, path selector to select an appropriate path for the packet, and load adaptor to dynamically estimate end-to-end delay on each path and adjust the allocation ratios accordingly.

To efficiently compute allocation ratios, the traffic splitter uses end-to-end delay estimates of the paths. The ratios are then re-adjusted by the load adaptor for every packet arrival. The main idea of E-DCLD is to decrease load on the path that has the longest delay and increase it on the path with the smallest delay. Load increase on the faster path is by the same amount of load reduced on the slower path. Even though E-DCLD can efficiently balance load across multiple links, it has similar drawbacks to EDPF. For instance, it ignores wireless path losses and congestion losses. Moreover, its work complexity increases with the number of paths since it requires every path to be monitored for traffic ratio adjustment when a new packet arrives.

Lin and Tsao [70] presented a dynamic bandwidth aggregation (DBA) scheduler, which resides in the network proxy. The objective of the scheduler is to enhance throughput by scheduling packets of the same stream over multiple links. The DBA scheduler architecture consists of two functional elements: traffic monitor and scheduler. The DBA traffic monitor observes traffic over

the wireless paths and feeds the information to the DBA scheduler to select a suitable link for the next packet. The scheduler computes the packet's expected departure time on each path, and similar to EDPF, the packet is scheduled on the path that offers minimum departure time. To adapt to the varying wireless link conditions, the departure time is calculated for each new packet arrival. The work complexity incurred to schedule a packet is, therefore, linear, and increases with the number of paths considered.

Ahmed et al. [71] proposed multi-server delay-budget ordered (MDO) scheduling architecture to schedule backlogged packets of an application over multiple links to the receiver. To ensure fairness among multiple applications (sessions), MDO uses some form of SRR. The MDO architecture includes a link monitor to estimate effective one way trip time on the link between the sender and the receiver. The effective one way trip time estimation is based on past observations. MDO uses effective one way trip time values to rank links that are available for use to achieve the desired bandwidth aggregation. Then, a packet that has the smallest delay tolerance (delay budget) is allocated to a path that offers the shortest trip time. This is a desirable feature for real-time applications, which usually need to be delivered to the receiver the soonest. The MDO scheduler attempts to schedule packets of the same flow on a single link, thus increasing the probability of in-order arrival. However, this may lead to inefficient load balancing when flow sizes are different. To adapt to varying link conditions, MDO performs link ranking in every scheduling round; but, like other latency-based multipath schedulers, it fails to capture important characteristics, such as channel losses.

Tsai et al. [72] developed Concurrent Multipath Transmission Scheme (CMTS) to distribute packets of a single application over multiple paths to enhance throughput while preserving the correct packet order. The main idea behind CMTS is to artificially equalize average delays of the available links by scheduling more packets on the faster link. To accomplish this, the average transmission interval time (TI_i) on the faster link is estimated. Let t_d denote end-to-end delay difference between the slow path and the fast path. The number of packets that can be scheduled on the fast link can be determined as

$$N_{fast} = \frac{t_d}{TI_i} + 1 \tag{6}$$

The total number of packets allocated to the paths in a scheduling round is given as

$$\sum_{i=1}^{N-1} N_{fasti} + 1 \quad (7)$$

where N is the total number of links. To ensure in-order delivery, the last packet from (7) should be allocated to the slowest link. However, the scheme has no mechanism to deal with the residual packet reordering at the receiver. Also, characterizing the wireless channel by latency alone and ignoring other important characteristics, such as wireless link losses, may not yield optimal gains.

Evensen et al. [73] introduced a multilink proxy to stripe IP packets of the same flow over multiple heterogeneous links, ensuring that the packets arrive in correct order while balancing the traffic load efficiently across the links. The proposed multilink proxy implements the functional elements: packet scheduler, delay equalizer, and path monitor. The packet scheduler determines traffic allocation ratios according to weighted round robin, where the weights are calculated based on link throughputs. The delay equalizer maintains a buffer to queue packets traversing the fast link so that they can experience similar delays to packets on the slow link. The total amount of time a packet on the fast link can be held in the buffer is bounded by the difference in delay between the fast link and the slow link. The packet scheduler and the delay equalizer rely on the path monitor for periodic updates on throughput and delay so that they can adapt allocation ratios and transmission schedules to network changes.

Even though the presented multilink proxy can achieve efficient load balancing and low reordering delay, it has some drawbacks. For instance, link throughput estimation is only based on available bandwidth, missing important channel characteristics, such as loss. This can lead to sub-optimal performance, especially for applications that are sensitive to loss. Furthermore, preventing packet reordering by throttling the transfer rate of a fast link is not efficient, as it reduces link utilization, thus leading to considerable throughput degradation and almost obviating the use of multiple links.

In [74], Evensen et al. improved their work in [73] to work with middleware network elements, such as network address translators (NATs). The proposed solution uses an IP tunnel between a multi-interface client and the proxy, thus creating a multi-path overlay network. IP packets are then transmitted to the client through the tunnels. To enable the solution to work with a NAT, NAT hole punching [75] can employed. To efficiently aggregate bandwidth over the multi-path overlay network, an adaptive packet scheduler is introduced to intelligently distribute packets across the network, ensuring that traffic is well balanced. The scheduler includes a congestion

control mechanism based on CCID2 [76], where the proxy periodically checks the links' congestion windows and assigns an incoming packet to a link with a larger congestion window. If no link has a congestion window that is big enough to accommodate the packet, it is dropped.

The scheduler algorithm is depicted in Fig. 3.6. The proposed bandwidth aggregation solution solely depends on the reorder buffer to correct packet reordering. Thus, a large buffer may be required to achieve efficient re-sequencing. However, as it was mentioned earlier, the implementation of a large buffer can yield long end-to-end delays.

```
1: max_capacity = MIN_VALUE
2: scheduled_link = None
3: links = [set of links with open congestion window]

4: if links == Empty then
5:      drop packet
6:      return None
7: end if

8: for all links do
9:      if capacity_link > max_capacity then
10:         max_capacity = capacity_link
11:         scheduled_link = link
12:     end if
13:end for
14:return scheduled_link
```

Fig. 3.6 Congestion window-based packet scheduler [74]

Mao et al. [77] presented an analytical framework for optimally splitting traffic over multiple paths. In the framework, traffic splitting is formulated as a constrained optimization problem. The objective is to minimize end-to-end delay, which includes path and re-sequencing delay. The incoming traffic stream is regulated by a (ρ, σ) leaky bucket, where ρ is long-term average rate, and σ is the maximum burst size of the stream. A closed-form solution to the optimization problem is derived. The solution provides optimal traffic split ratios that minimize end-to-end delay and conform to the leaky bucket parameters. However, the framework does not provide any mechanism at the sender to dispatch traffic units over the selected paths in correct order. Instead, it relies on the re-sequencing buffer at the receiver to correct packet reordering. Thus, it may require a large buffer to efficiently control packet reordering.

Zhong et al. [78] proposed an adaptive load balancing algorithm (ALBAM) whose objective is to achieve efficient utilization of the aggregated bandwidth capacity while reducing packet reordering. To select the best path for a packet, ALBAM lists all possible paths for incoming

packets; each path is associated with a quality value defined by

$$S_i = Q_i + V_i^* \tag{8}$$

where Q_i is queuing delay experienced by a packet on path i and V_i^* is the weight parameter on the path. V_i^* is used to configure traffic allocation ratios to conform to the desired application performance. When packets arrive, they are not scheduled immediately—rather, they are temporarily buffered to create a time gap between consecutive packets; this helps to control the transmission order. When scheduling resumes, a packet from the transmission buffer is allocated to a path that minimizes the quality value as defined by equation (8). To adapt to varying channel conditions, ALBAM monitors the paths and updates the quality value (S_i) periodically. ALBAM reduces packet reordering and enhances throughput, even for TCP applications. However, its use of transmission buffers to control the transmission order may introduce delays that may not be tolerated by real-time applications.

Manousakis et al. [79] presented INTELiCON, an intelligent connectivity framework for simultaneous use of multiple network interfaces to deliver high QoS in unreliable and resource limited networks. INTELiCON consists of four functional modules: Packet Processing, Decision, Measurements, and Control. The Packet Processing module implements transmission strategies, particularly round robin, to manipulate packets and transmit them dynamically across multiple network interfaces. Due to the use of round robin packet distribution, INTELiCON may not adapt well to variable wireless channel characteristics, thus leading to high packet reordering and unpredictable throughput and delay performance. To solve packet reordering, making higher layers oblivious to the use of multiple interfaces, INTELiCON maintains a reorder buffer to hold packets before delivering them to the application.

The Decision module uses an optimization algorithm to decide on the optimal connectivity strategy that the Packet Processing module should use to achieve efficient multipath communication. Varying traffic and network characteristics form an important set of input parameters to the optimization algorithm, thus enabling dynamic adjustment of the connectivity strategies to ensure adaptive multilink transmission. The Measurements module collects and distributes information about traffic and network conditions to all other modules. The information can be communicated using in-band and out-band methods, which are implemented by the Control module.

Link Layer: Koudouris et al. [43, 80] studied switched MRTD based on the GLL concept. Switched MRTD allows for transmission of user's data via one RAT at a time, switching between RATs with favorable characteristics any number of times during the transmission period. Switched MRTD is, therefore, a more adaptive form of MRTD. In [80], switching between the available RATs is triggered by throughput measurements. In every transmission interval, user's traffic is moved to a RAT with the highest supportable throughput. In [43], traffic is switched to a RAT with the shortest RTT. In both cases, loss rates are not considered. It is shown that switched MRTD achieves better performance in delay and goodput than naïve parallel MRTD. However, at medium traffic loads when the RATs show similar performance distributions, the performance of switched MRTD degrades considerably due to frequent switching and feedback information overheads. Consequently, further extensions are required to optimize switching and update frequency in order to enhance the performance of switched MRTD.

A similar approach to the GLL concept is studied by Kim et al. [81]. They introduced a cognitive convergence layer (CCL) to harmonize the different link layers' functions, thus creating a single virtual link layer interface between higher layers and the underlying link layer interfaces. To efficiently exploit the aggregated bandwidth, the CCL implements a traffic distribution policy at the sender and a reorder buffer at the receiver. The traffic distribution policy disperses traffic across the links in proportion to their available capacities. The available capacity is calculated as a function of link transmission time (LTT), which estimates the amount of channel occupancy time. The proposed multilink architecture relies only on the receiver to solve data reordering; thus, a large reorder buffer may need to be implemented.

The CCL-based multipath traffic distribution system is further studied in [82] using link delay as a metric to decide how data can be striped over tight-coupled WiMAX and WiFi networks. The traffic distributor calculates and adjusts traffic allocation ratios using link layer delay information that is periodically communicated to the sender by the receiver through a feedback mechanism. The solution can balance traffic load more efficiently than those that blindly disperse data units over the links. However, similar to [81], it solely relies on the reorder buffer to correct possible data reordering at the receiver and may require a large buffer to correct the reordering. Furthermore, assessing link quality based solely on delay information may not be optimal, as there are other important link characteristics, such as wireless path loss, which can affect performance of the link.

In [83], air-time cost based traffic splitting over tight-coupled WiMAX and WiFi networks is presented. Air-time cost is defined as a measure of channel occupancy time for a single packet transmission. Cumulative air-time cost (CAC) is, therefore, total channel occupancy time for packet transmission on a link, and it is determined by recording the total number of packets transmitted through link *i* during a time interval (T_i). CAC is normalized to available channel time and used to compute traffic allocation ratios. A flowchart in Fig. 3.7 summarizes the procedure to determine traffic allocation ratios. The proposed air-time cost-based model is reported to achieve more rapid adaptation to link variations than RTT-based model. However, more attributes, such as loss and delay, should be incorporated to further improve performance.

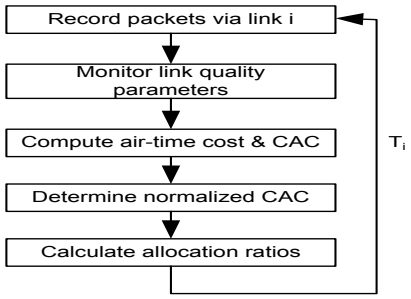

Fig. 3.7 Air-time-based traffic splitting [83]

3.4 Summary of Bandwidth Aggregation Solutions

This chapter has presented a comprehensive and critical review of bandwidth aggregation solution from the literature. In this section, a summary of the received bandwidth aggregation solutions is provided.

3.4.1 Summary of Bandwidth Aggregation Solutions

The reviewed bandwidth aggregation solutions are classified based on their ability to adapt to changing traffic and network path conditions. The solutions that configure their traffic allocation ratios and distribution policies to match varying traffic and network characteristics are classified as adaptive, while those that are based on static configurations are called non-adaptive. The solutions are further classified according to the layer of the network protocol stack at which they perform traffic striping, that is, they are classified into application, transport, network and link layer solutions. Table 3.2 is a summary of non-adaptive bandwidth aggregation solutions. The non-

adaptive bandwidth aggregation solutions can achieve load balancing to some extent. However, when traffic and network conditions vary rapidly, the load balancing ability may diminish significantly.

Also, non-adaptive approaches mostly rely on the reorder buffer implemented at the receiver to correct reordering; so, to efficiently solve data reordering, a large reorder buffer may be required. A reorder buffer that is too large can increase the packets' end-to-end delays, thus affecting performance of delay sensitive applications. Moreover, a buffer that is too large may not be practical for handheld devices with limited resources. Despite lack of adaptation to traffic and network conditions, non-adaptive bandwidth aggregation solutions have low computational complexity and communication overhead, so they can be easy to implement.

A summary of adaptive bandwidth aggregation solutions is depicted in Table 3.3. Adaptive bandwidth aggregation solutions rectify most of the deficiencies of non-adaptive solutions. However, this may come with increased complexity. The complexity may result from the communication that is required to periodically gather information about traffic and network conditions. It can also be due to finding the best path from a list of the available paths for each packet. Most of the solutions in Table 3.3 use a single metric to adapt to traffic and network dynamics. Therefore, such solutions may not be efficient in a practical scenario where the network may be characterized by multiple variables.

Also, very few adaptive solutions deal with reordering at both the sender and the receiver; most approaches implement reordering solution at either the sender or the receiver. Furthermore, almost all the reviewed solutions do not incorporate cross-layer design to allow communication between the layers of the network protocol stack. The load balancing problem is addressed by all the solutions. How efficient the solutions can handle load balancing depends on a number of factors. For instance, packet-level solutions can achieve higher load balancing efficiency than flow-level approaches. Feedback information reporting time scale can also affect load balancing efficiency. That is, when the information is reported to the sender in a short time scale, capacity variations that can warrant reconfiguration of allocation ratios can be captured soon enough to prevent load imbalance. Long time scale reporting, on the other hand, may miss important capacity variations and incur a drop in load balancing efficiency.

Table 3.2 Summary of Non-adaptive Bandwidth Aggregation Solutions

Approach	Striping Point	Load Balancing	Reordering Solution	Adaptation Metric	Cross-layer Information	Computational Complexity
MuniSocket [35]	Applicat;ion	Yes	Receiver-based	None	No	-
MPTCP [36]	Transport	Yes	Receiver-based	None	No	-
WRR [39]	Network	Yes	None	None	No	O(1)
SRR [26]	Network	Yes	Receiver-based	None	No	O(1)
Kaspar et al. [40]	Network	Yes	Receiver & Sender	None	No	O(n)
Kim et al. [41]	Network	Yes	None	None	No	O(1)
GLL [42]	Link	Yes	None	None	No	-

Table 3.3 Summary of adaptive Bandwidth Aggregation Solutions

Approach	Striping Layer	Load Balancing	Reordering Solution	Adaptation Metric	Cross-layer	Complexity
Luo et al. [44]	Application	Yes	No	Delay	No	O(1)
LPS [45]	Application	Yes	No	Loss	No	O(1)
FPS [45]	Application	Yes	No	Loss	No	O(1)
Xue et al. [48]	Application	Yes	No	Bandwidth	No	O(1)
Horder [49]	Application	Yes	No	Delay; loss	No	O(n)
Kaspar [51]	Application	Yes	No	Bandwidth	No	-
W-SCTP [52]	Transport layer	Yes	No	Delay	No	O(n)
LS-SCTP [54]	Transport layer	Yes	Receiver-based	Bandwidth	No	O(1)
cmpSCTP [55]	Transport layer	Yes	Receiver-based	Bandwidth	No	O(1)
ATLB [56]	Transport layer	Yes	Sender & receiver	Delay	No	O(n)
pTCP [57]	Transport layer	Yes	Receiver	Bandwidth	No	O(1)
cmpRTCP [58]	Transport layer	Yes	Receiver-based	Bandwidth	No	O(1)
FPS [59]	Transport	Yes	No	Bandwidth	No	-
FP-cross-layer [60]	Transport	Yes	No	Bandwidth	Yes	-
MTCP [25]	Transport layer	Yes	Sender-based	Delay	No	-
EDPF [24, 62]	Network layer	Yes	Sender-based	Delay	No	O(n)
Liu [64]	Network layer	Yes	Sender-based	Delay	No	O(n)
TS-EDPF [65, 66]	Network layer	Yes	Sender-based	Delay	No	O(n)
PT [67]	Network layer	Yes	Sender & receiver	Delay	No	O(n)
E-DCLC [69]	Network layer	Yes	Sender-based	Delay	No	O(n)
DBA [70]	Network layer	Yes	Sender-based	Delay	No	O(n)
MDO [71]	Network layer	Yes	Sender-based	Delay	No	O(n)
CMTS [72]	Network layer	Yes	Sender-based	Delay	No	-
Evensen et al. [73]	Network layer	Yes	Sender-based	Bandwidth	No	O(1)
Mao et al. [77]	Network layer	Yes	Receiver-based	Delay	No	O(n)
ALBAM [78]	Network layer	Yes	Sender-based	Delay	No	O(n)
INTELiCON [79]	Network layer	Yes	Receiver-based	Delay; loss	No	O(1)
Koudouridis [43]	Link Layer	Yes	Receiver-based	Delay	No	O(n)
Koudouridis [80]	Link Layer	Yes	Receiver-based	Bandwidth	No	O(n)
Kim et al. [81]	Link Layer	Yes	Receiver-based	Delay	No	O(1)
Kim et al. [82]	Link Layer	Yes	Receiver-based	Delay	No	O(1)
Kim et al. [83]	Link Layer	Yes	Receiver-based	Air-time	No	O(1)
Nightingale et al. [47]	Application Layer	No	Sender-based	Bandwidth	No	O(n)

Chapter 4

A Network Layer Solution for Combating Packet Reordering in Concurrent Multipath Transmission

Bandwidth aggregation for concurrent multipath transmission promises performance improvement for multi-homed devices in heterogeneous networks. Concurrent multipath transmission is accomplished by striping packets from the same application across multiple networks simultaneously. The performance of concurrent multipath transmission, however, can be affected adversely by packet reordering, which occurs when packets belonging to the same application traverse different network paths and arrive at the receiver out of the correct order. To combat packet reordering and achieve efficient concurrent multipath transmission, efficient striping mechanisms need to be developed.

This chapter presents a new concurrent multipath transmission solution, which incorporates a packet striping mechanism that works at the network layer to schedule packets from the same application for transmission across multiple network paths. The proposed striping solution schedules packets across the network paths based on the paths' end-to-end delay to ensure that the packets arrive in the correct order, thus minimizing the reorder buffer requirements at the receiver. Experimental results have shown that the proposed concurrent multipath transmission scheme achieves less packet reordering and reorder buffer requirements than the round-robin based concurrent multipath transmission solutions.

The rest of the chapter is organized as follows. Section 4.1 describes a multipath packet scheduling model, which plays a critical role for combatting packet reordering in concurrent multipath transmission. Section 4.2 details the operation of a novel multipath packet scheduler that has been proposed to intelligently and adaptively stripe application packets across multiple network paths simultaneously. Section 4.3 proposes a mechanism that handles residual packet reordering at the receiver in order to circumvent the reorder buffer blocking problem. Section 4.4 presents experimental evaluation of the proposed concurrent multipath transmission solution. A critical analysis of the experimental results is also provided. Section 4.5 summarizes the chapter.

4.1 Multipath Packet Scheduling Model

Fig. 4.1 illustrates the proposed multipath packet scheduling model to stripe packets of a single application flow across multiple heterogeneous network paths. The multiple network paths are used between the sender and the receiver to overcome transmission bottleneck of a single path, thus scaling up capacity for applications, such as high definition video streaming, with high bandwidth demands. The multipath packet scheduling algorithm is implemented at the sender to accomplish parallel packet transmission over multiple network paths. The receiver retrieves packets from its multiple network interfaces, which correspond to the network paths, on a first-come-first-serve basis and reassembles the packets into a single packet flow that can be consumed by the application. The proposed scheduling process is detailed as follows:

- The multipath packet scheduler resides at the sender node, which is essentially an interworking element, such as a network proxy connected to the different paths as shown in Fig. 4.1. The sender stores packets in a common buffer where the scheduler reads and stripes them across multiple network paths.
- Each path maintains transmission buffer to temporarily store the packets before de-queuing them into the respective link.
- The scheduler distributes the packets across the various paths based on a quantum that is predetermined according to the paths' estimated delays. The estimated delay for path i is denoted by d_i. d_i is estimated for each path using the receiver reports as specified by the RTCP framework [84]. Path delay can also be determined using packet-par probing.
- The scheduling mechanism schedules lower sequence number packets on faster paths to ensure that they can get to the receiver before higher sequence number packets on the slower network paths. Thus, the correct packet sequence can be maintained at the receiver.
- At the receiver, packets are retrieved from the interfaces sequentially, starting from the fastest path to the slowest. The packets that arrive at the receiver out of the anticipated order are buffered until the reordering is resolved.

The main objectives of the proposed multipath packet scheduling mechanism are as follows: (1) to properly balance the packet load across the available network paths based on their delay performance; (2) to preserve the original packet sequence when the packets arrive at the receiver.

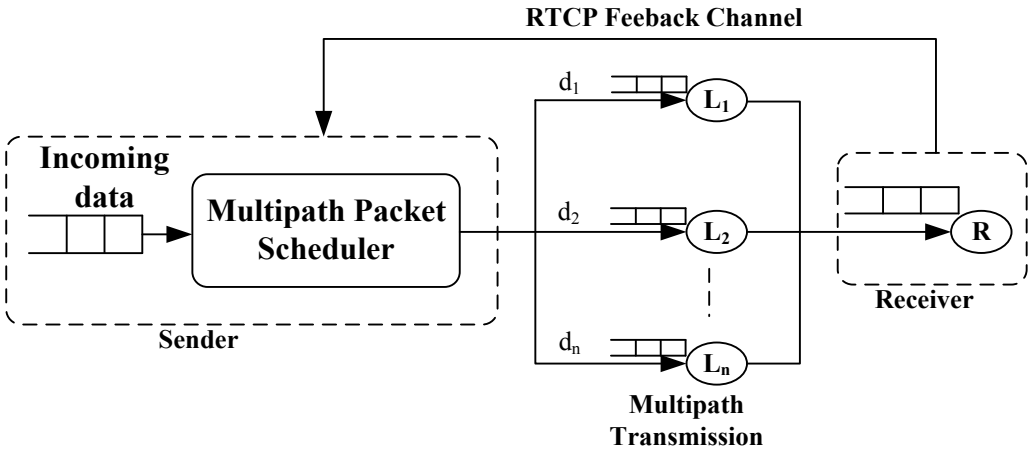

Fig. 4.1 Multipath Packet Scheduling Model for a Single Packet Flow

To design an efficient multipath packet scheduler, a number of important questions need to be answered. Firstly, how much traffic load should be assigned to each of the paths to yield the desired load balancing performance? Secondly, in what order should the packets be dispatched to the different paths to preserve the original packet sequence at the receiver? Finally, what is the impact of variable length packets on the proposed scheduling process? To answer these questions, first, a quantum is derived for each path based on its delay performance relative to the slowest path. The quanta are assigned such that the paths' total delays in a scheduling round can be equalized, which is important for minimizing packet reordering. To further reduce packet reordering, the proposed scheduler allocates lower sequence number packets on faster paths to arrive at the receiver before higher sequence number packets. For any consecutive variable size packets on different paths, a simple mechanism is devised to ensure that the lower sequence number packet arrives at the receiver in the anticipated order to avoid packet reordering. The answers to the questions are discussed in great details in the rest of the chapter. Even though the striping point for the proposed solution is the network layer, signalling to gather control information is a cross-layer approach.

4.2 The Proposed Multipath Packet Scheduling Algorithm

A multipath packet scheduler should balance traffic across the network paths properly, ensuring that the paths are utilized according to their capabilities. Moreover, the scheduler must assign the packets across the different paths for concurrent multipath transmission such that the correct packet sequence can be preserved at the receiver. To balance traffic across the paths, the

proposed multipath packet scheduler derives appropriate traffic quantum for each path based on estimated network paths' end-to-end delay performance. Intuitively, the slower (high delay) network paths are allocated small quanta—with the slowest network path getting the smallest quantum. This can help to avoid excessive delays that could result from overloading underperforming network paths. The smallest quantum is set to the maximum allowable packet (P_{max}) size to ensure that at least one packet can be dispatched to the slowest network path in a scheduling round, thus achieving work-conservation, which guarantees utilization of all the aggregated network paths.

Definition 1.1 A *scheduling round* is a packet allocation cycle, which begins with the network paths having their unused ('fresh') quanta and ends when the quanta have been exhausted or when there are no more packets (bytes) to send. In a scheduling, the scheduled packets are transmitted simultaneously across the different network paths.

Let σ denote the smallest quantum that can be assigned to a network path. For any two network paths, L_i and L_j, with the respective estimated delays, d_i and d_j, where $d_i < d_j$, the proposed scheduler will assign σ worth of bytes to L_j as its quantum in a scheduling round since the path is the slowest. When some of the application packets are smaller than σ, the path's minimum quantum may serve more than one packet. That is, for packets P_k and P_{k+1} with smaller sizes than σ, if $\rho \geq P_k + P_{k+1}$, it is permissible for the packets to account as the minimum quantum on the same path. The proposed scheduling mechanism balances traffic across the paths based on estimated delay performance. In a scheduling round, the traffic load is well balanced if the paths' total delays are equal. To achieve this, the delay equalizing factor α_{ji} is used to derive the quantum of the faster path, L_i; in fact, α_{ji} is the $d_j : d_i$ ratio, which is determined as

$$\alpha_{ji} = \frac{d_j}{d_i} \tag{1}$$

From equation (1), it can be deduced that path L_i is α_{ji} as fast as path L_j. Hence, to equalize the total delays between the paths, whenever σ bytes are allocated to the slower path, L_j, $\alpha_{ji}\sigma$ bytes should be assigned to the faster path, L_i. Consequently, the quantum of path L_i is $\alpha_{ji}\sigma$ bytes in a scheduling round. Maximum byte count that can be transmitted in a scheduling round is simply the sum of the paths' quanta. For N paths, the maximum byte count, C, in a scheduling round is given as

$$C = \sigma(\sum_{i}^{N-1} \alpha_{ji} + 1) \quad (2)$$

Equalizing the paths' delays (i.e. reducing the paths' delay difference), as the proposed multipath packet scheduler attempts to do, has been shown to reduce packet reordering [40]. To further reduce packet reordering, the packet arrival precedence constraint is set between consecutive packets traversing different network paths. That is, the higher sequence number packet must arrive at the receiver after or at the same time as the lower sequence number packet. The proposed scheduler enforces the set precedence constraint by dispatching the lower sequence number packet for transmission on the faster path so that it can arrive at the receiver before the higher sequence number packet on the slower path. To illustrate, consider a stream of n packets of P_{max} size, with sequence numbers 1 through n. When $\alpha_{ji}\sigma = k, k < n$, packets are scheduled in order on L_i, the $(k+1)^{th}$ packet must be scheduled on L_j. When the load is allocated properly according to the paths' quanta, the k^{th} packet on L_i and the $(k+1)^{th}$ packet on L_j should arrive at the receiver at the same time. Thus, the set packet precedence constraint will be satisfied and packet reordering can be mitigated.

Example 1.1. Consider three P_{max} size packets, 1, 2 and 3 ready to be scheduled on L_1 with latency of 1 time unit and L_2 with latency of 2 time units. According to equation (2), the delay equalizing factor between the paths is $\alpha_{21} = \frac{2}{1} = 2$. Therefore, the quantum for L_1 is $\alpha_{21}\sigma = 2\ P_{max}$ size packets, while L_2 can transmit $\sigma = 1\ P_{max}$ size packet in a scheduling round. To obey the packet precedence constraint, the lower sequence number packets must be scheduled on the faster path to precede higher sequence number packets on the slower path. Therefore, the allocation for the two paths in the scheduling round is: $L_1 = \{1, 2\}$ and $L_2 = \{3\}$. When transmission on the links begins at the same time, packet 1 on L_1 will arrive at the receiver before packet 3 on L_2. Packets, 2 and 3, will get to the receiver at the same time, thus conforming to the packet precedence constraint and, therefore, avoiding packet reordering. When packets arrive at the receiver at the same time, it is proposed that they should be retrieved from the interfaces, starting from the fastest interface to the slowest. The significance of reading arriving packets from the interfaces in this manner will be highlighted in the next section.

The packet precedence relations established between the faster paths and the slower paths form a special matrix called the unit lower half matrix, where the elements are given as

$$a_{ji} = \begin{cases} 1, & j = i \\ 0, & j < i \\ int\left(\frac{d_j}{d_i}\right) - \sum_{j=i}^{j-1} int\left(\frac{d_j}{d_i}\right), & j > i \end{cases} \quad (3)$$

a_{ji} in the matrix is the number of packets that should be scheduled on L_i relative to L_j, considering all the previous allocations on L_i in a scheduling round. For n paths, $L_1, L_2, \ldots, L_n$, where L_1 is the fastest path and L_n is the slowest path, the corresponding packet precedence relations in a scheduling round are given by the matrix in Table 4.1.

Table 4.1 Packet Precedence Relations for n Network Paths

j,i	L_1	L_2	L_3	L_4	...	L_n
L_1	1	0	0	0	...	0
L_2	a_{21}	1	0	0	...	0
L_3	a_{31}	a_{32}	1	0	...	0
L_4	a_{41}	a_{42}	a_{43}	1	...	0
...	...	...	...	...	...	0
L_n	a_{n1}	a_{n2}	a_{n3}	a_{n4}	...	1

The matrix depicted in Table I is interpreted as follows. Consider allocating traffic load to path L_1 and path L_2. Before allocating a packet at $(i = 2, j = 2)$ to L_2, $a_{21} + 1$ packets need to have been allocated to L_1 to ensure in-order arrival at the receiver. Similarly, before allocating a packet to L_3 at $(i = 3, j = 3)$, $a_{32} + 1$ packets need to have been allocated to L_2, and $a_{31} + a_{21} + 1$ packets need to have been allocated to L_1 to maintain the desired sequence in the arriving packet stream. As mentioned previously, the allocation process goes on until there are no more packets to schedule or the paths have exhausted their quanta. Algorithm I puts together the ideas of the proposed multipath packet scheduler.

The computational complexity of the proposed scheduling process is $O(nlogn)$, where n is the number of network paths. This complexity comes from sorting the available paths in the increasing order of the estimated delays. When the network paths are sorted, assigning a packet to a network path can be performed in constant time. In practice, the path sorting complexity is not anticipated to be significant since the number of interfaces an end host can use simultaneously may be considerably low. Link states updates in the proposed scheduling framework are performed at the end of every scheduling as opposed to per-packet updates. There is a trade-off between path conditions update frequency and the accuracy of the scheduling mechanism. Calculating link conditions for every out-going packet can increase network traffic load, thus decreasing available bandwidth for the application and, therefore, reducing achievable throughput in the multipath transport chain. Moreover, the changes in path conditions may be transient, making per-packet monitoring an overkill that could lead to lack of stability. On the other hand, infrequent updates may affect the accuracy of the scheduler and consequently lead to some performance loss. On these notes, the proposed scheduler will work best when path conditions variability does not warrant per-packet updates.

Algorithm I Multipath Packet Scheduling Algorithm

Require: Active set A; queue (Q) of packets ready to schedule.
$a_{ji} \leftarrow 0$; // outstanding packets to schedule on path i relative to path j;
$n \leftarrow |A|$; //Sorted number of paths available in a scheduling round
$countPkts_i \leftarrow 0$; //count packets on path i
for $i \leftarrow 1$ to n
    for $j \leftarrow 1$ to n
        if $i == j$ **then**
            Allocate P_k to path i;// k[th] packet in the ready queue
            $countPkts_i \leftarrow countPkts_i + 1$;
        else if $i < j$ **then**
            $a_{ji} = \frac{d_j}{d_i} - countPkts_i$;
            Allocate next a_{ji} packets to path i;
            $countPkts_i \leftarrow countPkts_i + a_{ji}$;
        else
            break;
        end if
    end for
end for

4.3 Addressing Residual Packet Reordering at the Receiver

Even though the proposed multipath packet scheduling algorithm can tackle packet reordering efficiently, there may still be residual packet reordering at the receiver, which must be corrected to avoid performance degradation. The receiver deals with the residual packet reordering by maintaining a reorder buffer to hold out-of-order packets. When packets arrive, the receiver runs a re-sequencing algorithm, which retrieves the packets from the different network interfaces in a first come first serve basis. The packets that do not match the expected sequence number are held in the reorder buffer, waiting to be released by a lower sequence number packet that is yet to arrive. For real-time applications, the reordered packets can only be kept in the reorder buffer as long as their playback deadlines can be met. The packets whose playback deadlines expire before packet reordering is resolved are cleared from the reorder buffer since they can no longer be useful to the application.

In the case of simultaneous packet arrivals at the receiver, a re-sequencing algorithm that retrieves packets from the interfaces in a random order can suffer performance loss. Consider the situation in Fig. 4.2, where the free space in the reorder buffer can only accommodate two reordered packets. Assume that packets 3-6 have arrived at the same time and must be retrieved from the different interfaces. The packets are expected in the order: '3, 4, 5, 6.' Reading the packets from if3 to if0 will use up all the free space in the reorder buffer since the first two packets (6 and 5) read are not the expected ones, and they must be buffered until the lower sequence number packets arrive to release them. With the reorder buffer full, packet 4 cannot be accommodated, and it must, therefore, be discarded as illustrated in Fig. 4.2. When packet 4 is dropped, even after retrieving packet 3, packets 5 and 6 may still remain in the buffer, thereby continuing to block new arrivals in the reorder buffer.

To circumvent the problems caused by random retrieval of arriving packets, a retrieval mechanism is added to the re-sequencing process to prioritize the reading of the expected sequence numbers during simultaneous arrivals. The proposed mechanism predicts which interface is likely to have the expected packet based on the nature in which the packets are scheduled for transmission at the sender. The proposed scheduler at the sender dispatches packets to the different network paths such that the lower sequence number packets traverse the faster paths. Therefore, reading from the fastest path to the slowest can improve the chances of retrieving lower sequence number

packets before higher sequence number packets. For instance, reading from if0 to if3 avoids the use of the reorder buffer, which is desirable. In the case of a miss, that is, the packet read is not the expected one, the proposed retrieval method moves the packet into the reorder buffer. Information about the estimated performance of the paths in a scheduling round can be communicated to the receiver using the established RTCP system. The packet retrieval method described here is illustrated by Algorithm II.

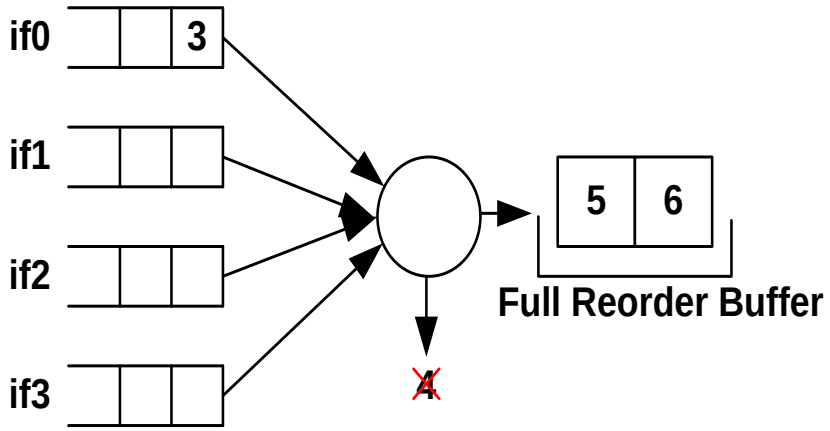

Fig. 4.2 Packet drop due to full reorder buffer

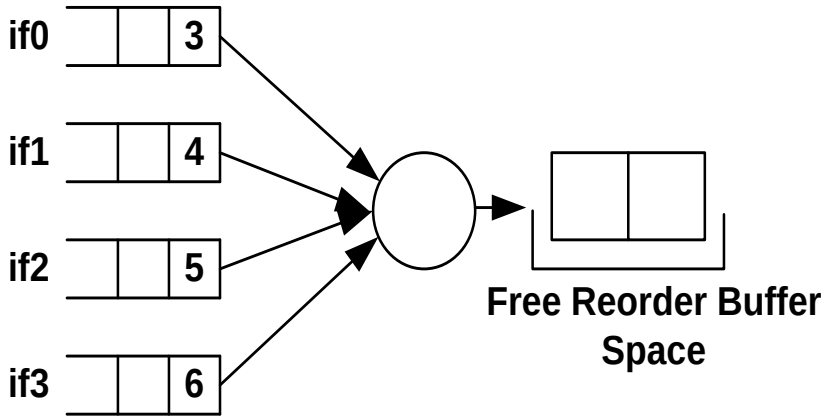

Fig. 4.3 Simultaneous Arrivals at the Receiver

Algorithm II Re-sequencing Algorithm

```
1:  if packets arrive simultaneously on n interfaces then
2:      for i ← lowestIndex to n(highestIndex)
3:          retrieve packet from i;
4:          if packet is reordered then
5:              hold it in the buffer for re-sequencing;
6:          else
7:              deliver packet to higher layers;
8:          end if
9:      end for
10: else if one packet at a time
11:     retrieve the arriving packet;
12:     perform step 4-8;
13: end if
```

Example 2. Consider a stream of nine equal-length packets (1 to 9) ready to be scheduled simultaneously over three network paths, L_1, L_2 and L_3, with latencies of 1, 2 and 3 time units, respectively. Packet precedence constraints between the packets are enforced by striping the packets across the network paths according to the matrix depicted in Table 4.2. The packets are scheduled over the paths in two scheduling rounds. According to equation (2), we have: $int(delay(L_2)/delay(L_1)) = int(2/1) = 2$. This tells us that two packets on L_1 should precede a packet to be scheduled on L_2. Similarly, a packet scheduled on L_3 is preceded by three packets on L_1. The relationship between L_2 and L_3 is such that a packet on L_3 should be preceded by one packet on L_1. When the packets arrive at the receiver, they are retrieved according to Algorithm II. For instance, packet 1 is expected to get to the receiver first, and it will be sent to higher layers without any problems of packet reordering. Packets, 2 and 3 are expected to arrive at the same time. In that

case, the algorithm dictates that a packet on the lower index interface must be retrieved first; so, packet 2 will be read before packet 3 and there will be no packet reordering experienced. The same procedure is followed to read packets, 4 and 5 from the interfaces to avoid packet reordering.

Table 4.2 Precedence Matrix for the given example

j,i	L_1	L_2	L_3
L_1	*1*	*0*	*0*
L_2	*1*	*1*	*0*
L_3	*1*	*0*	*1*

4.4 Performance Evaluation of the Proposed Concurrent Multipath Transmission Solution

This section presents undertaken performance evaluation of the proposed concurrent multipath transmission solution. The evaluation compares the proposed multipath transmission approach to concurrent multipath transmission schemes, which use round robin to stripe application packets simultaneously across multiple network paths. The round robin concurrent multipath transmission has been illustrated in Chapter 3. The evaluated solutions have been implemented and incorporated into the ns-3 [85] simulation environment. The objective of the evaluation is to ascertain the efficacy of the proposed solution in combatting packet reordering. As explained earlier, packet reordering occurs when packets belonging to the same flow arrive at the receiver out of the expected order.

The performance of the evaluated concurrent multipath solutions has been measured in terms of reordering delay, reordering entropy and reorder buffer overflow. Reordering delay measures the time required to recover from packet reordering, i.e., the time a reordered packet spends in the reorder buffer until the expected lower sequence number packet arrives. Reorder entropy has been explained in Chapter 2. Reorder buffer overflow measures the number of packets lost when the reorder buffer overflows because of increased packet reordering.

4.4.1 Evaluation Procedure and Network Topology

Fig. 4.4 and Fig. 4.5 show the network topologies considered for the simulation of the investigated concurrent multipath transmission schemes. Two multipath network configurations have been used in the simulation experiments: two-path and three-path networks. For the two-path multipath network environment, delay heterogeneity between the paths has been created by varying the delay on Path2. The resulting delay ratios for the paths are: 1:1, 1:2, 1:3, …, 1:10. In the three-path multipath environment, the delay on Path2 has been fixed at twice the delay on Path1, while the delay on Path3 is varied to increase the delay heterogeneity. The resulting delay ratios on the three-path configuration are: 1:2:1, 1:2:2, 1:2:3, 1:2:4, 1:2:5, …, 1:2:10. It is desirable for an efficient concurrent multipath transmission solution to adapt to the differences in delay into order to mitigate packet reordering for improved performance.

In the simulation, the sender (server) sends 1500-byte packets to the destination (client) over multiple network paths. The packets have been generated according to a video trace of a video sequence. The simulation experiments have been carried for infinite reorder buffer and finite reorder buffer configurations. The results presented here only serve as proof concept of concurrent multipath transmission under an efficient multipath packet scheduler such as the one proposed in the thesis. In the discussion of results, the proposed solution is simply referred to as 'Proposed' while the round robin solution is 'Round Robin'.

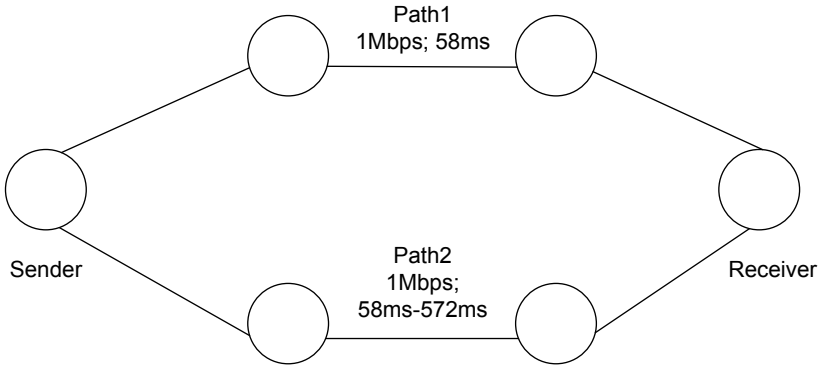

Fig. 4.4 Simulation Topology for 2-path scenario

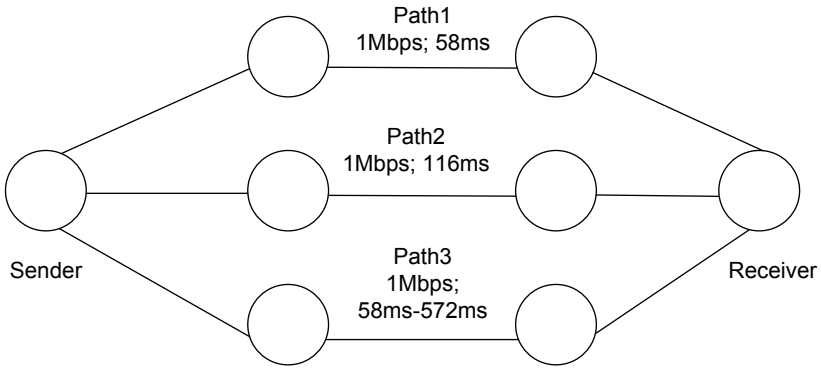

Fig. 4.5 Simulation Topology for 3-path scenario

4.4.2 Delay Heterogeneity with infinite reorder buffer

This experiment investigates the behaviour of the proposed and the round robin concurrent multipath transmission solutions under heterogeneous delays and infinite reorder buffer. With infinite reorder buffer, the reordered packets are not lost but kept in the buffer until packet reordering is resolved. This helps to fully capture the reordering delay performance of the different concurrent multipath transmission solutions across all the reordered packets. A finite reorder buffer would result in some of the reordered packets discarded and their reordering delay not captured.

4.4.2.1 Two-Path Multipath Scenario

For the two-path concurrent multipath transmission scenario, propagation delay on Path1 is fixed at 58ms for the duration of the simulation, while the delay on Path2 is varied from 58ms to 572ms, which creates increasing delay heterogeneity between the paths. Bandwidth on all the network paths is fixed at 1 Mbps. Fig. 4.6 shows average reordering delay experienced in the two-network multipath scenario under infinite reorder buffer. For the two solutions—Proposed and Round Robin, reordering delay increases as the delay on Path2 increases. The increasing delay on Path2 increases the delay difference between the paths, thus resulting in increased packet reordering. When packet reordering is high, reordering delay—the time required to recover from the reordering—is long. Reordering delay under the proposed concurrent multipath transmission solution, however, increases more gradually and it is much lower than in the case of Round Robin concurrent multipath transmission. This is because the Proposed solution is aware of the delay difference between the different network paths and it attempts to stripe the packets across the network paths such that the lower sequence number packets are scheduled on the network path

that will deliver them to the receiver before higher sequence number packets. This effectively removes or significantly lowers packet reordering, which results in small reordering delays.

The Round Robin concurrent multipath transmission experiences higher reordering delays because the packets are dispersed across the network paths in a cyclic manner without considering the network paths' delays. With the Round Robin concurrent multipath transmission, the higher sequence number packets on the faster network path always arrive at the receiver before the lower sequence number packets on the slower path, which results in packet reordering. The Packet reordering for the Round Robin increases more rapidly as the delay heterogeneity increases, which results in higher reordering delays. The reason for the rapid increase in packet reordering for Round Robin is due to it poor load balancing capability. Round Robin distributes the packets evenly across the network paths, and the packets on the slower paths are quickly overtaken those on the faster network path, which exacerbates the disorder at the receiver.

High reordering delay degrades application throughput performance for both TCP and UDP applications. For TCP applications, reordering delay triggers false fast retransmissions and throttling of the congestion window, thus reducing the expected throughput. For UDP applications that are time-sensitive, packet reordering increases the packets' end-end delays, making it highly likely for the packets to miss their playback deadlines, which is not desirable. Against this background, the proposed concurrent multipath transmission scheme can deliver better performance for both UDP and TCP applications since it is able to reduce reordering delay significantly.

Fig. 4.7 shows reordering entropy under the Proposed and the Round Robin concurrent multipath transmission solutions. Reorder entropy measures the packet disorder in the received packet sequence. High reorder entropy indicates severe packet reordering with large packet displacements. The Proposed solution achieves significantly lower reorder entropy than the Round Robin, and the entropy increases gradually as opposed to the sharp increase observed under the Round Robin multipath transmission. The sharp increase in reorder entropy under the Round Robin is indicative of severe packet reordering and large packet displacements. The gradual increase in reorder entropy under the Proposed solution indicates that the solution can effectively reduce packet reordering to achieve improved concurrent multipath transmission performance.

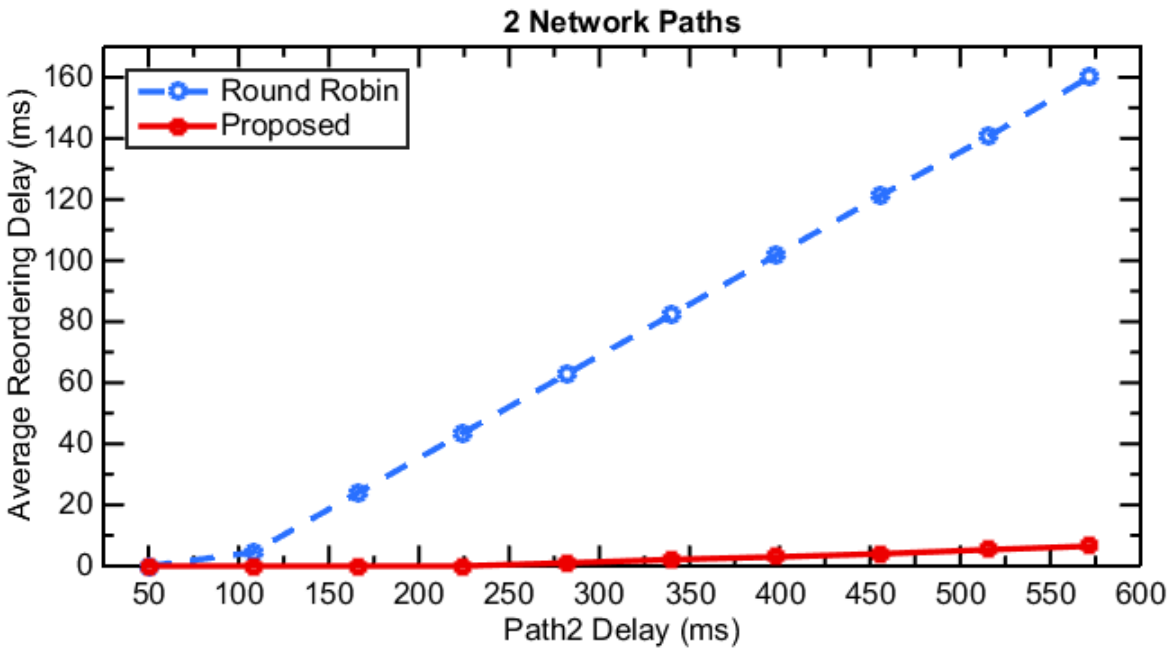

Fig. 4.6 Reordering Delay on Two-Path Multipath Transmission

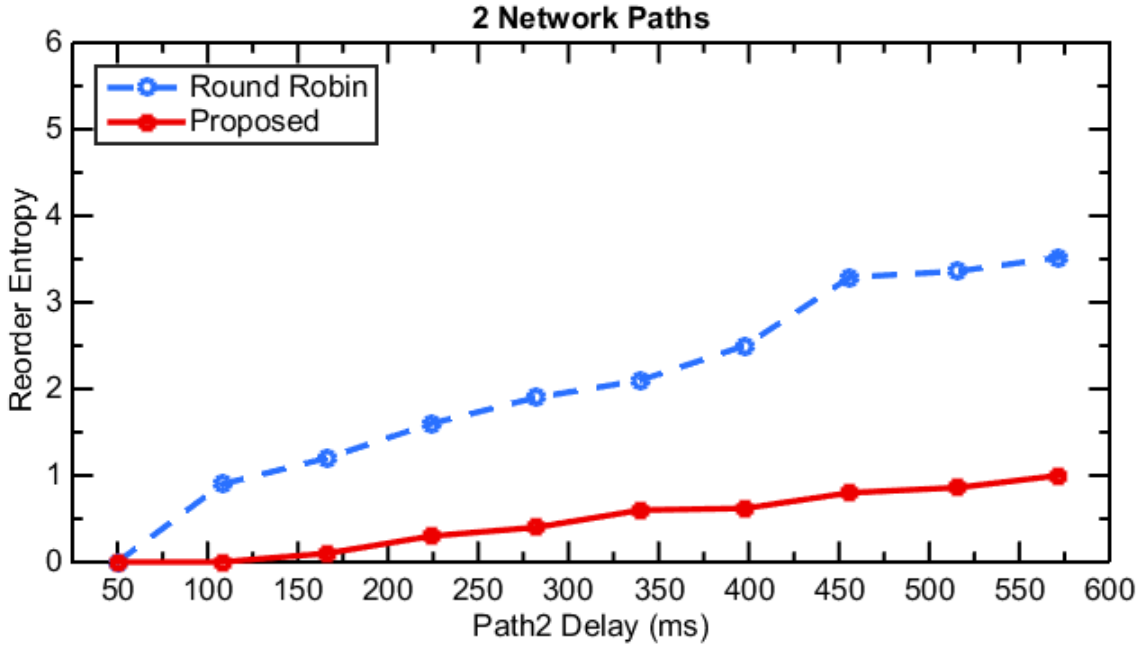

Fig. 4.7 Reorder Entropy on Two Paths with Heterogeneous Delay

4.4.2.2 Three Network Path Scenario

In this part of the simulation experiments, three network paths have been set up to enable concurrent multipath transmission between the sender and the receiver. The objective is to investigate the impact of increasing the number of network paths on concurrent multipath transmission under the Proposed and the Round Robin solutions. The delay settings for the network paths are: 58ms ol2n Path1, 116ms on Path2 and 58ms-572ms on Path3. The expectation is that packet reordering will increase with the increase in the number of network paths as it was observed in [19], thus resulting in higher reordering delay than in the case of two network paths. It is desirable for an efficient concurrent multipath transmission solution to not only adapt to the delay heterogeneity between the network paths but the increase in the number of network paths forming a multipath network environment.

Fig. 4.8 depicts the reordering delay results for the three-path concurrent multipath transmission under the two investigated solutions (Proposed and Round Robin). Compared to the two-path multipath transmission scenarios, reordering delay experienced under both the Proposed and the Round Robin concurrent multipath transmission solutions has worsened. This confirms the results reported in [86] that packet reordering increases with the number of network paths that have heterogeneous delays. The packet reordering is further exacerbated by the increase in delay heterogeneity resulting from the delay on Path3 being varied from 58ms to 572ms. High packet reordering results in long reordering delays. The longer reordering delays for the three-path multipath transmission are indicative of the packet reordering that is higher than in the two-path scenario.

Even though it has been shown that reordering delay is higher in the three-path case, the increase in reordering delay under the Proposed solution for the three-path scenario has not increased considerably, which attests to the Proposed solution's capability to adapt to varying network conditions (delay heterogeneity and number of network paths). For the Round Robin, the reordering delay has increased considerably, showing that adaptation to changing network conditions is key to accomplishing efficient concurrent multipath transmission. The lesson learned is that there may be a threshold to the number of network paths required for a beneficial concurrent multipath transmission. Beyond the threshold, concurrent multipath transmission may not bring about any benefit. It is not in the scope of this thesis to determine such a threshold.

Fig. 4.9 shows comparison of the Proposed and Round Robin solutions using reorder entropy in the three-path concurrent multipath transmission scenario. As expected, it can be observed that the reorder entropy has also increased with the number of network paths in much the same way as the reordering delay. The reorder entropy has not increased noticeably for the Proposed solution but has increased significantly for the Round Robin. This can be attributed to the scheme's ability to adapt to network conditions in order to curb the disorder and the degree of packet displacements in the received packet sequence.

An adaptive solution such as the Proposed solution leads to low and controlled reorder entropy, whereas a non-adaptive solution such as the Round Robin experiences high reorder entropy that spikes out of control as the number of network paths increases. The lesson learned in the experiment is that adaptation to path heterogeneity is critical for improved concurrent multipath transmission performance. Adaptation to path heterogeneity will certainly be even more important in wireless networks, which are often characterized by highly varying characteristics.

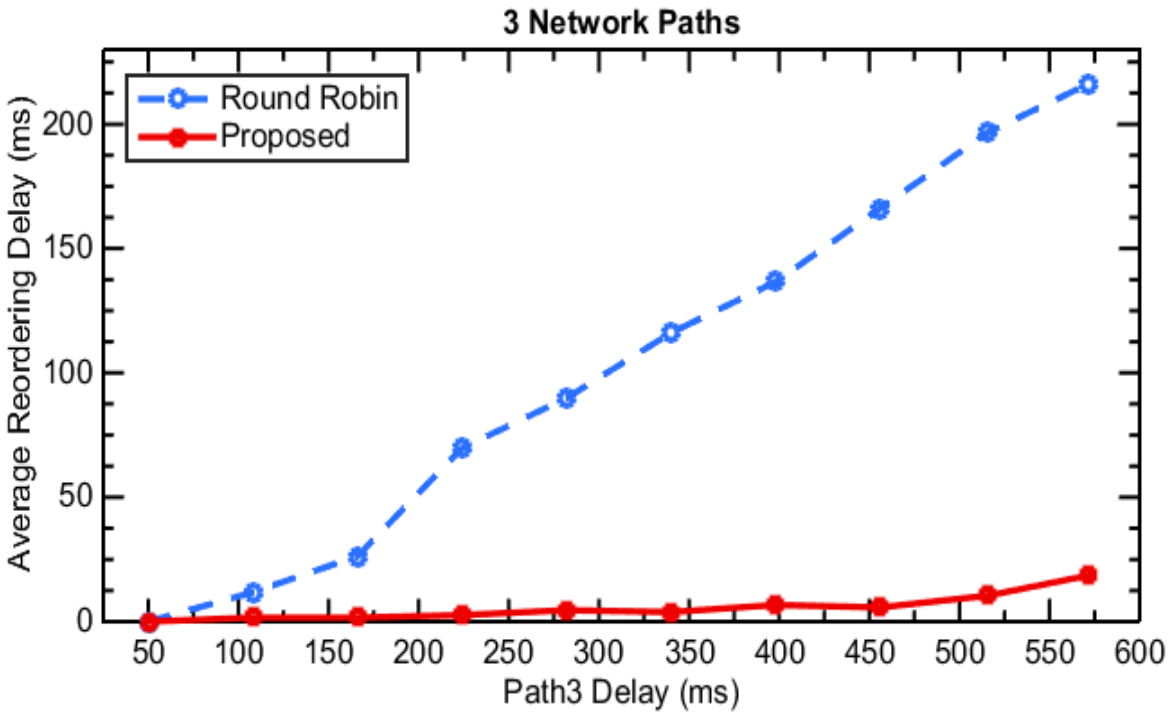

Fig. 4.8 Reordering Delay on Three Paths with Heterogeneous Delay

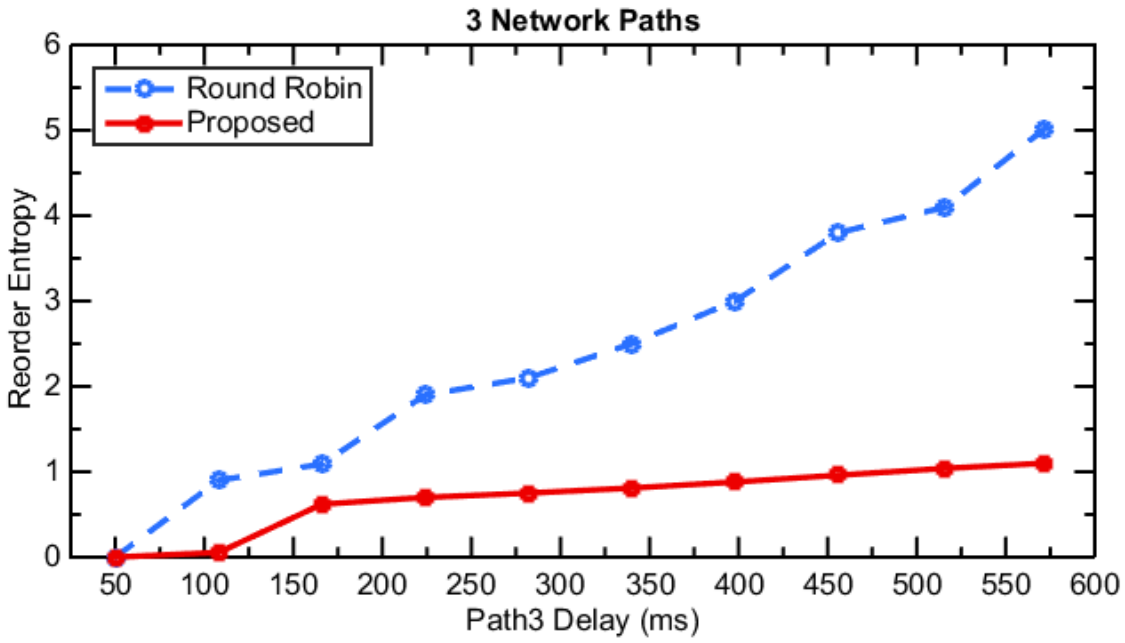

Fig. 4.9 Reorder Entropy on Three Paths with Heterogeneous Delay

4.4.3 Delay Heterogeneity with finite reorder buffer size

The previous experiments were based on an infinite reorder buffer, which always has space to keep a packet that arrives out of sequence. In practice, storage—especially for handheld devices—is limited, which results in the use of reorder buffers with finite size. The objective of the simulation experiments with finite reorder buffer size is to study the impact of packet reordering on the performance of a reorder buffer with finite capacity. The performance of the two investigated solutions—the Round Robin and the Proposed—has been analysed in terms of reorder buffer overflow, which measures the percentage of reordered packets that get discarded due to a full reorder buffer (reorder buffer blocking).

The problem of reorder buffer blocking has been illustrated in Fig. 4.2. A large number of reorder buffer blocking events results in increased packet loss, which degrades overall throughput performance. It is, therefore, imperative to minimize reorder buffer blocking for improved throughput in concurrent multipath transmission.

4.4.3.1 Two and Three Network Paths with Reorder Buffer Size of 3 Packets

In this simulation scenario, the reorder buffer size is set to 3 packets, where a packet is 1500 KB. The settings may not be as they would be in practice, but they suffice as proof of concept to show how packet reordering and a finite reorder buffer at the receiver can affect the performance of concurrent multipath transmission in heterogeneous networks.

Fig. 4.10 and Fig. 4.11 show the reorder buffer overflow results when the reorder buffer size is set to 3 packets for the two-path multipath and the three-path multipath, respectively. In Fig. 4.10, as expected, the Round Robin, which has been shown to suffer long reordering delays due to high packet reordering, has the highest reorder buffer overflow percentage. The high packet reordering under the Round Robin results in a large number of packets that must be buffered until the packet reordering can be corrected. When the reorder buffer is finite (3 packets in this case), it quickly overflows, and most of the reordered packets have to be discarded, thus causing high reorder buffer overflow. The Proposed solution, on the other hand, suffers much lower reorder buffer overflow. This is because the solution is able to reduce packet reordering significantly, resulting in a very small number of packets that need to be buffered to correct the packet reordering. The reorder buffer under the solution proposed in the thesis seldom overflows, which is indicative of the solution's ability to mitigate packet reordering effectively.

For the three-path concurrent multipath transmission, it was shown earlier that the disorder (reorder entropy) and the reordering delay were worse than in the two-path multipath scenario. This is because packet reordering experienced in a two-path multipath transmission is lower than in the three-path scenario. The higher packet reordering in the three-path scenario would result in a larger number of packets that need to be stored in the reorder buffer in order to resolve packet reordering. For the reorder buffer that is constrained to 3 packets, most of these packets will be blocked and discarded, which leads to a more severe reorder buffer overflow. Fig. 4.11 confirms that—for the same reorder buffer size (3 packets in this case)—the three-path concurrent multipath transmission suffers worse reorder buffer overflow than the two-path scenario. This holds mostly for the Round Robin concurrent multipath transmission as the reorder buffer overflow has barely increased under the Proposed solution—thanks to the solution's ability to adapt to the increase in the number of network paths in order to curb packet reordering, thus avoiding the risk of overflowing the reorder buffer. The disorder under the Round Robin has been shown to increase

significantly with the number of network paths. Since the increased disorder causes an increased number of packets that must be stored in the reorder buffer, the constrained reorder buffer leads to increased overflow, which results in high packet loss.

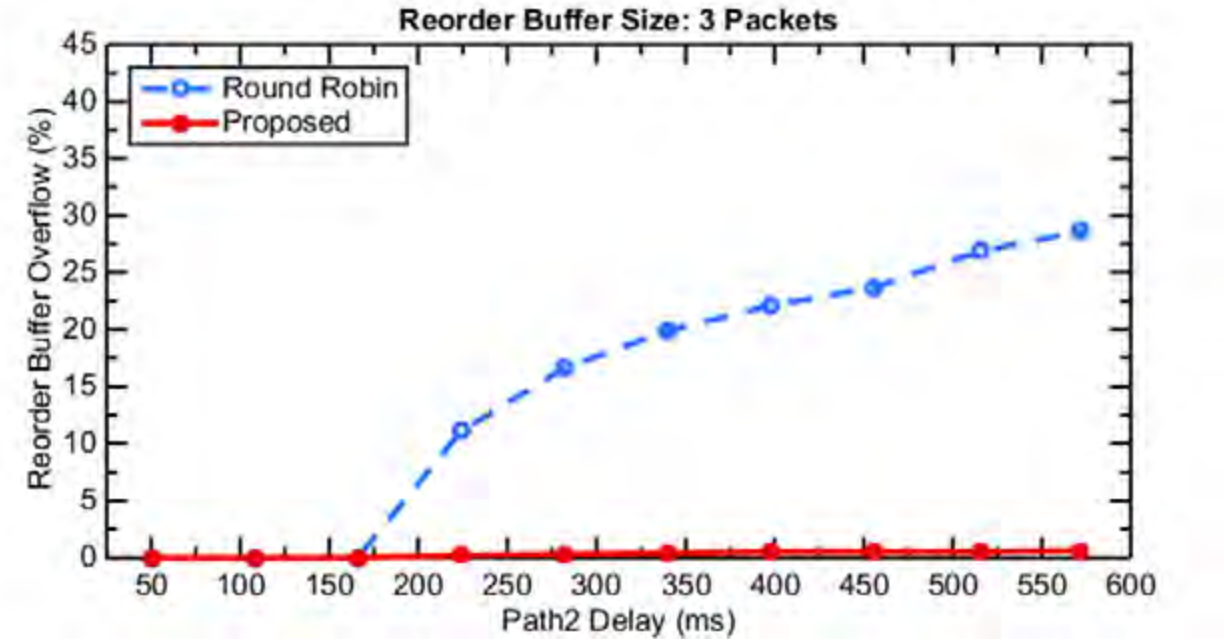

Fig. 4.10 Reorder Buffer Overflow on Two Network Paths with Finite Reorder Buffer

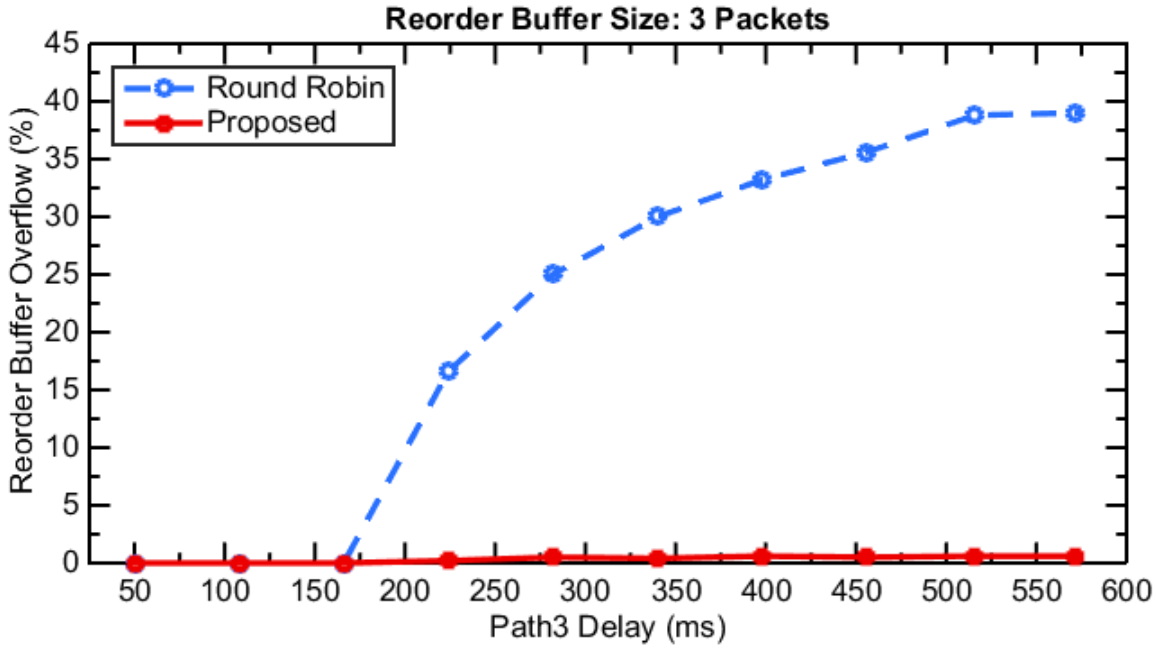

Fig. 4.11 Reorder Buffer Overflow on 3 Network Paths with Finite Reorder Buffer

4.4.3.2 Two and Three Network Paths with Reorder Buffer Size of 4 Packets

In this simulation experiment, the reorder buffer size has been set to 4 packets. Fig. 4.12 and 4.13 show the reorder buffer overflow results in the two-path and three-path concurrent multipath transmission cases, respectively. As expected, a large reorder buffer can lead to decreased packet loss as there is more space to store reordered packets to avoid severe reorder buffer overflow. However, the reorder buffer overflow in the three-path scenario is still higher than in the two-path multipath scenario as it has been the case in the previous simulation experiments. This is due to the high packet reordering experienced in the three-path case.

The reorder buffer overflow under the Proposed solution has barely changed compared to the experiments with the reorder buffer set at 3 packets. This attests to the Proposed concurrent multipath transmission solution's robustness and ability to adapt to different network conditions in a multipath transmission environment. The Round Robin, on the other hand, seems to be favoured by a larger reorder buffer size as the reorder buffer overflow experienced in the 4-packet reorder buffer is significantly less than in the 3-packet reorder buffer case. However, the Round Robin still performs much worse than the Proposed solution in all cases.

The lesson learned from the simulation experiments with finite reorder buffer size is that a reorder buffer that is too small can lead to high packet loss, which can severely degrade application throughput performance. This is especially true for solutions, such as the Round Robin, which are prone to high packet reordering. A larger reorder buffer can minimize packet loss due to reorder buffer overflow, which can improve throughput performance. However, this can be at the expense of high reordering delay, which can degrade performance of delay-sensitive applications. Optimal reorder buffer settings must therefore take into consideration the type of application in question. For instance, delay-insensitive applications that demand high throughput may benefit from a reasonably large reorder buffer, whereas real-time applications may benefit from a reorder buffer that is small enough to curb average reordering delay. It was also observed that reorder buffer overflow is higher for a larger number of network paths and it gets worse as the delay heterogeneity between the network paths increases. Therefore, the optimal reorder buffer settings must also consider the number of network paths and the delay heterogeneity. Small delay difference between the network paths may warrant small reorder buffer, while large delay difference may require a large reorder buffer.

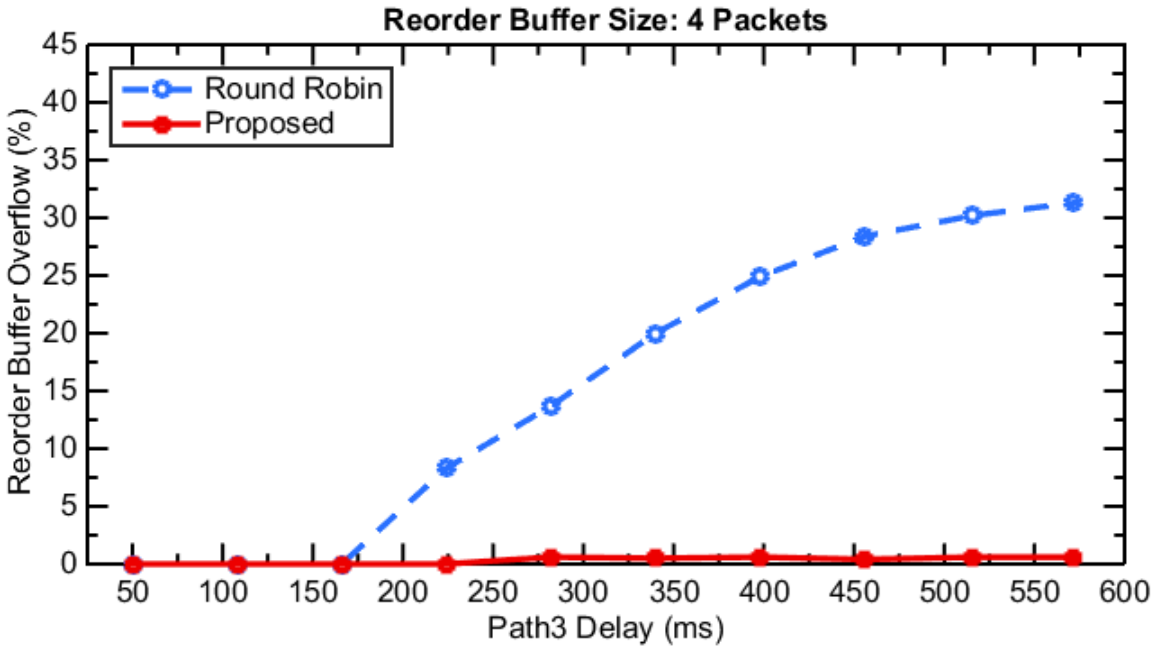

Fig. 4.12 Reorder Buffer Overflow on 3 Network Paths with Finite Reorder Buffer

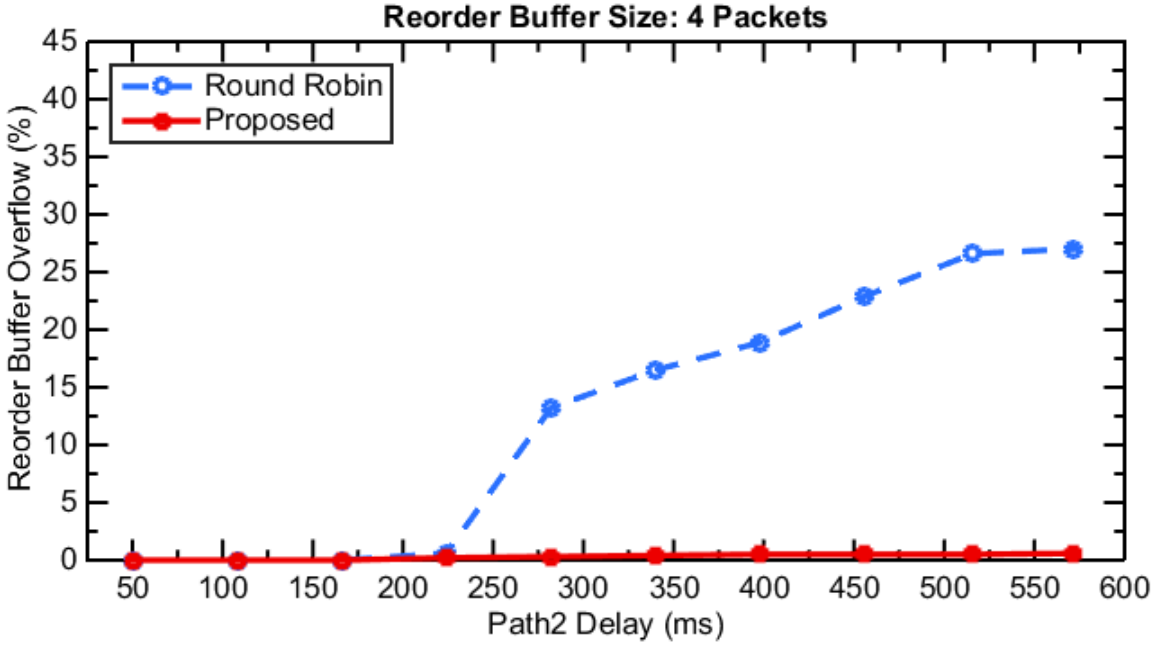

Fig. 4.13 Reorder Buffer Overflow on 3 Network Paths with Finite Reorder Buffer

4.5 Chapter Summary

To address packet reordering in concurrent multipath transmission, a new network layer solution, which includes an adaptive multipath packet scheduler, has been presented in this chapter. The proposed multipath packet scheduler stripes packets of the same flow over multiple network paths taking into consideration the paths' delay heterogeneity. The multipath packet scheduler ensures that—whenever possible—the lower sequence number packets arrive at the receiver before the higher sequence number packets, which effectively and significantly reduces packet reordering.

The performance of the proposed solution has been analysed in the ns-3 simulation environment using several multipath network scenarios. The experimental results have shown that the proposed concurrent multipath transmission system can better decrease the disorder in the packet sequence (reorder entropy), average reordering delay and reorder buffer overflow, which characterize packet reordering.

Chapter 5

Concurrent Multipath Transmission for H.264 Scalable Video Coding

This chapter presents the design and evaluation of a new concurrent multipath transmission framework for H.264 Scalable Video Coding in a heterogeneous network environment with multi-homed user's devices. At the core of the proposed framework is a video packet scheduling mechanism, which is called a video packets distributor (VPD) in this thesis. The video packets distributor adaptively stripes SVC encoded video packets across multiple heterogeneous network paths to a multi-homed user's device, which receives the video packets through multiple network interfaces. The proposed concurrent multipath streaming framework for H.264 SVC has been evaluated in the ns-3 simulation environment using real video traces under various network conditions. The results show that the framework is a workable solution for improving H.264 SVC streaming quality in bandwidth-limited networks, where no single network path may have enough bandwidth to meet the target video bitrate. The proposed multipath video streaming solution has only been tested for video-on-demand, which allows users to stream high quality prerecorded video. Evaluation of the framework for interactive video streaming is planned for future work.

The rest of the chapter is structured as follows. Section 5.1 provides an overview of the proposed multipath video streaming framework, which includes a discussion of the functions performed by the different components that make up the proposed multipath video streaming system. In Section 5.2, video packets distributor, which is the core of the proposed multipath video streaming solution is discussed in detail. Section 5.3 presents a comprehensive performance evaluation of the proposed multipath video streaming system, which includes a detailed discussion of the proposed evaluation framework, performance metrics and the experimental scenarios used to establish the efficacy of the proposed concurrent multipath transmission of on-demand H.264 SVC encoded video. The experimental results are discussed in Section 5.4, where performance of the proposed multipath video streaming solution is compared with two reference schemes. Finally, Section 5.4 provides the summary for the chapter.

5.1 The Proposed Multipath Video Streaming Framework

The proposed framework is illustrated by the network architecture in Fig. 5.1. The framework enables concurrent multipath transmission of on-demand H.264 SVC encoded video for multi-homed devices. The network elements that make up the framework include a multi-homed user's device, a network proxy, SVC video streaming server and multiple network paths connecting the multi-homed user's device to the network proxy. The network proxy is some kind of interworking gateway enabling the integration of the different access networks that a multi-homed user's device can connect to. The integration realized by the network proxy is loose coupling. Loose coupling integration allows for the different access networks to operate independently and also to be under different administrative domains. With loose coupling, a multi-homed device can improve performance by aggregating bandwidth from multiple network interfaces supported by different network providers.

The user's device accesses the services of the video streaming server via the network proxy. The user's device and the network proxy are both multi-homed, thus allowing for concurrent multipath transmission to be configured between them. Each network path between the user's device and the network proxy is characterized by available bandwidth, end-to-end loss rate and end-to-end delay. When the user's device makes a request to the video streaming server for a video-on-demand service, the network proxy is configured to intercept the request and forward it to the streaming server on behalf of the device. Upon receiving a reply from the streaming server, the network proxy appropriately processes the reply to send to the user's device via multiple network paths whenever necessary. It should be noted that in the event a single network path has enough bandwidth to meet the required video bitrate, concurrent multipath configuration is not necessary.

The presented concurrent multipath video transmission proposal, as highlighted earlier, is premised on a scenario where no single network path in a heterogeneous network has enough bandwidth to meet the required video bitrate. The network proxy uses the Internet Engineering Task Force (IETF) real time streaming protocol (RTSP) [87] to set up a video streaming session between the user's device and the video streaming server. The video traffic from the video streaming server to the user's device is transported as the real-time transport protocol (RTP) [84] data units over the User Datagram Protocol (UDP). Video streaming over the transmission control

protocol (TCP) is outside the scope of the work presented in the thesis.

The video packets distributor, which is implemented by the network proxy, must periodically get up-to-date information about the network paths' changing conditions (delay, loss rate and bandwidth) so that the concurrent multipath transmission of the RTP-based video packets can be adapted accordingly to avoid possible performance degradation. In this research, a network path monitoring mechanism, which is based on the real-time control protocol (RTCP) is used to periodically communicate the status of the network paths between the user's device and the network proxy. Specifically, the research in this thesis uses the IETF RTCP extension for concurrent multipath transmission [88].

The RTSP/RTP/RTCP-based interaction between the multi-homed user's device, the network proxy and the video streaming server in the proposed multipath streaming process is illustrated in Fig. 5.2. First, the network proxy and the user's device set up RTSP sessions over multiple network paths to enable concurrent multipath transmission. The multi-homed mobile device and the network proxy agree on the number of network paths in the concurrent multipath transmission configuration using periodic network interface advertisements and connectivity checks. The network path between the network proxy and the streaming server is assumed to be stable and it has large enough bandwidth capacity to meet the requirements of a video-on-demand service; this is a valid assumption since the bottleneck is often expected to be between the network proxy and the user's device. As a result, the network proxy and the video streaming server communicate over a single RTSP session and therefore a single flow of RTP packets.

A successful setup of the RTSP sessions between the different network elements in the proposed system is followed by a flow of RTP packets to the multi-homed user's device. It is critical for the flow of RTP packets to be adapted to variations in the network paths' conditions, which are quite common in wireless networks. Such adaptation helps to avoid performance degradation, which could result from unbalanced load and transmission of the important parts of the video traffic on the network paths that experience high loss rates. The adaptation process in the proposed multipath video streaming chain relies on periodic RTCP reports to determine changes in the network paths' characteristics so that appropriate reconfiguration of the packet striping decisions can be made. The details of the functions implemented by each of the network elements (multi-homed user's device, network proxy and video streaming server) of the proposed

concurrent multipath transmission framework are discussed in detail below. Fig. 5.4 depicts the components that encapsulate the different functions that make up the proposed multipath streaming system.

At the streaming server, H.264 video encoding is implemented to generate suitable bit-streams for the requested video. The network proxy implements several functional components such as the session manager for RTSP session establishment and management, the video packets distributor to stripe the video packets across multiple network paths for concurrent multipath transmission, and the RTCP-based network path manager to estimate the network paths' characteristics in terms of delay, loss rate and throughput. On the multi-homed user's device, a re-sequencing unit to reassemble video packets from multiple network paths is implemented. The multi-homed user's device also has an RTCP-based network path manager to gather and communicate network performance statistics to the network proxy.

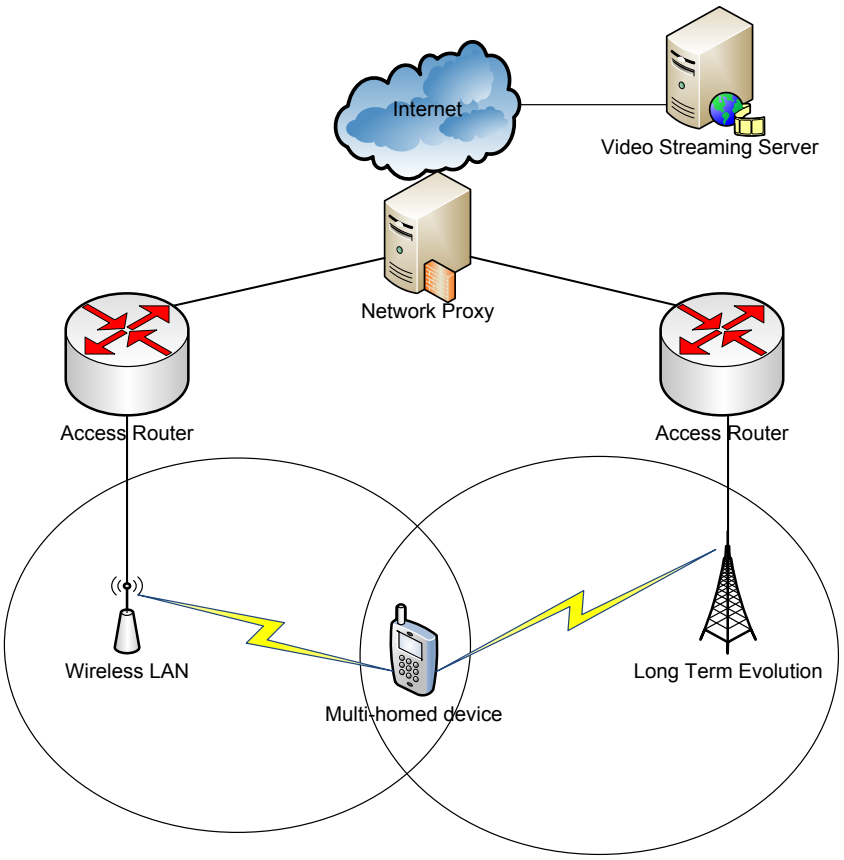

Fig. 5.1 Framework for Multipath Streaming of H.264 SVC encoded Video

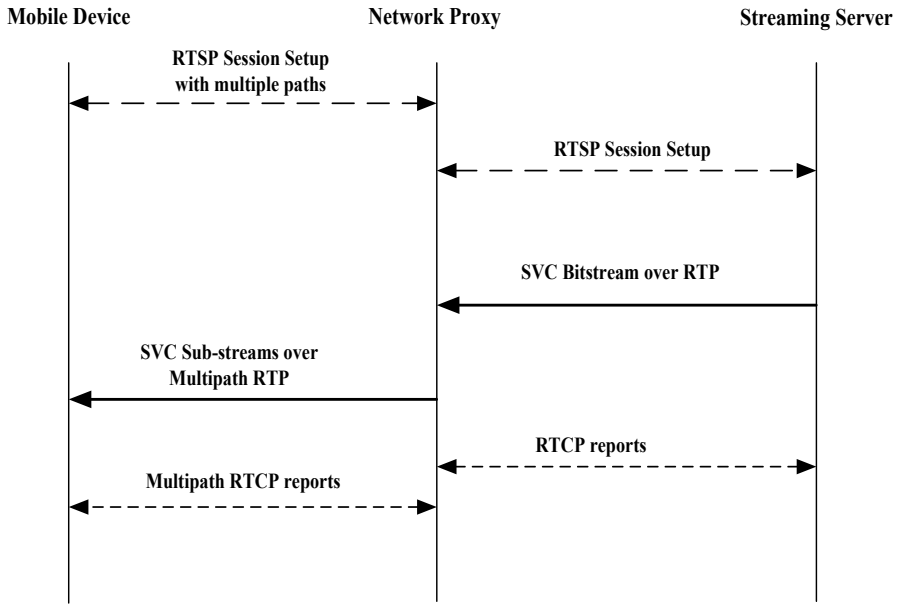

Fig. 5.2 Interaction between user's device, network proxy and streaming server

5.1.1 The Streaming Server Functions

The streaming server encodes the requested video sequence based on the H.264 SVC format. The resulting bit-stream is sent as RTP payload to the network proxy to forward to the consumer device. The SVC format comprises three types of scalability: spatial, temporal and quality [89]. Spatial scalability represents the video sequence in multiple resolutions. Temporal scalability, on the other hand, makes it possible to have multiple frame rates for a given resolution. For quality scalability, the video sequence is encoded at different quantization levels, thus resulting in multiple quality resolutions. Each unique partition in frame rate or resolution is often referred to as a layer. The lowest layer is called the base layer (BL), while the rest of the layers are known as enhancement layers (EL).

The base layer is compliant with the non-scalable H.264/AVC, so it can be decoded even by legacy devices that do not support the SVC extension. The different layers in the SVC video sequence are characterized by hierarchical dependencies, where the layer above (l_i) needs information from the layer below (l_{i-1}) for decoding. The only layer that is self-contained and can be decoded independently is the base layer. Therefore, the lower layers must be received before higher layers to enable efficient decoding; so, the lower layers are more important and must be transmitted with a higher priority than higher layers. Fig. 5.3 illustrates the SVC interlayer and

intra-layer dependencies. The arrows emanate from the reference frames and point to the dependent frames. Dependent frames require the reference frames for decoding, thus making the reference frames more important. The video transmission and adaptation process must therefore assure the highest priority transmission of the reference frames to avoid decoding errors, which could lead to significant video distortion.

The SVC layer data is stored in transport units called network abstraction layers (NALUs), which can easily be transmitted over a network. The NALUs are encapsulated as RTP packets to enable synchronization between the sender and the receiver. The different NALUs representing the different SVC layers make up the SVC bit-stream. The bit-stream can be truncated to extract specific sub-streams matching the available resources and specific video application quality needs. This is mostly desirable in the wireless network environment with time-varying channel characteristics and hand-held user's devices with heterogeneous capabilities such as resolution, memory and processor speeds.

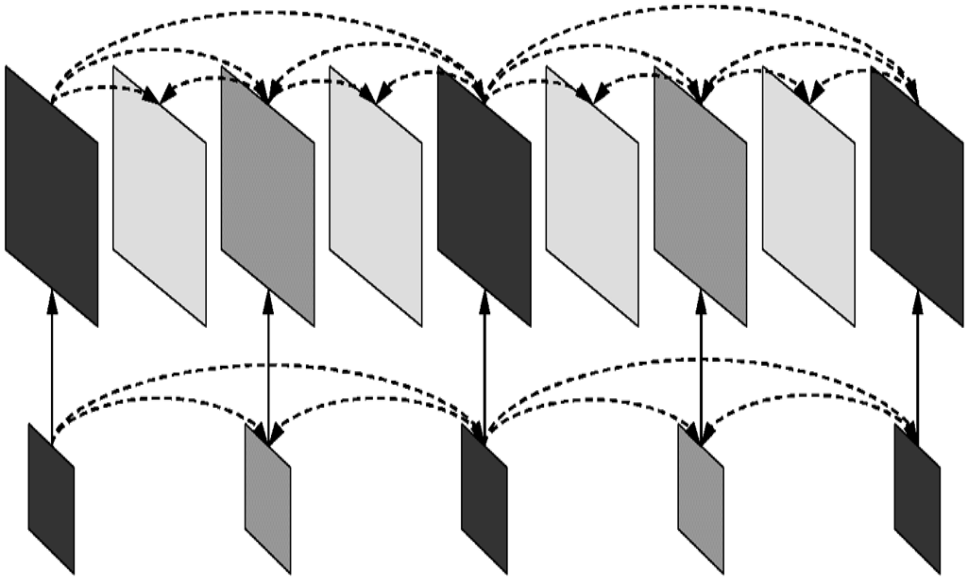

Fig. 5.3 Multi-layer H.264 SVC Structure with interlayer predictions

5.1.2 The Network Proxy Functions

The network proxy implements three functional components in the proposed multipath streaming system: the session manager, the video packets distributor (VPD) and the RTCP-based

network path manager. The session manager enables the network proxy to perform RTSP-based session setup transactions with the mobile device and the streaming server. When the multi-homed user's device sends a video session setup request to the streaming server, the network proxy intercepts the request transaction and appends appropriate header information to make the request on behalf of the mobile device. The session setup response from the streaming server is processed by the network proxy, preparing the response to traverse multiple network paths to the user's device. The set of network paths between the network proxy and the user's device is dynamically maintained by adding new network paths and removing the stale network paths. Addition of new network paths would usually be done before the session setup stage and it can also be done during the on-going video session to scale up capacity for the video call. The problem of discovering the network paths to recruit for concurrent multipath transmission can be resolved by using network path discovery specifications such as the IETF Interactive Connectivity Establishment (ICE) [90]. The stale network paths can be relegated from the candidate network paths for concurrent multipath transmission using information from the RTCP network performance reporting system.

The session manager is also responsible for media retrieval from the video streaming server. It then uses a bit-stream extraction function to extract a suitable SVC sub stream based on the multi-homed user's device's capabilities in the form of computation and resolution and also the bandwidth capacity pooled from the different network paths. The information about the mobile device's capabilities can be communicated using the session description protocol (SDP) [91] during the RTSP session setup.

In addition to session management, the network proxy needs to perform network path monitoring to get up-to-date status of the different network paths in order to perform the necessary adaptations to avoid video streaming performance degradation. Information about the status of the paths is gathered through the RTCP-based path manager. The path manager gets performance reports about individual network paths from the RTCP feedback from the user's device. The feedback information from the device is mainly RTCP receiver reports about the network paths' end-to-end delay, end-to-end packet loss rate, delay jitter, etc.

At the core of the proposed multipath SVC encoded video streaming chain is a Video Packets Distributor (VPD), which is also implemented by the network proxy. The VPD intelligently and adaptively stripes the video packets from the extracted bit-stream over multiple

network paths for concurrent multipath transmission. The VPD uses performance information from the network proxy's network path manager to forecast video packets' delivery times on the different network paths, and the packets are scheduled on the paths that can meet the required playback and loss performance. The objective of the proposed VPD is to ensure that the video packets can be delivered to meet their playback as well as preventing the loss of high priority video packets by scheduling them on the network paths that have the lowest loss rates. Avoiding the loss of high priority video data is crucial for minimizing drifting errors, which can significantly degrade SVC video quality performance. The VPD operation is adaptive, that is, the video packets striping decisions are adjusted dynamically to reflect changes in the network paths' status. A more detailed account of the design and analysis of the VPD—which is the most important component of the proposed multipath video streaming framework—is given in Section 5.2.

5.1.3 The Multi-homed User's device Functions

On the user's device, a re-sequencing unit (RU) is implemented to read arriving video packets from the user's device's multiple network interfaces. The RU uses a buffer to store the video packets until they are reassembled and sent to an appropriate video application. In the event packets with unresolved decoding dependencies miss their playback, the RU discards them to make room for the video packets that are within their decoding deadlines. The user's device is also equipped with the receiver RTCP-based network path manager to measure quality of the received video stream in terms of delay, loss, delay jitter, etc. Then, a receiver report, detailing performance information across the different network interfaces on the multi-homed user's device, is compiled and sent to the network proxy over the established RTCP feedback channel.

The frequency at which the user's device sends performance reports to the network proxy should be carefully set to enable the network proxy to get an accurate estimate of the paths' status. Inaccurate path status can lead to video packets striping decisions that can hurt video streaming performance. For instance, during rapidly varying network path conditions, the reporting frequency could be set to be short enough to correctly capture the variability of the paths' conditions. The RTCP reporting frequency can be increased or reduced within the 5% limit of available bitrate [92]. If the recommended 5% limit is inadequate, it can be increased for more frequent reporting. The reporting frequency can also be increased by using smaller RTCP packets.

In this work, the user's device's network path manager is configured to send the performance reports to the network proxy at the end of the transmission of a Group of Pictures (GoP). This allows the network proxy to schedule the video packets of the next GoP based on the network paths' characteristics estimated from the previous GoP transmission. However, this remains configurable depending on the variability of the network paths' conditions.

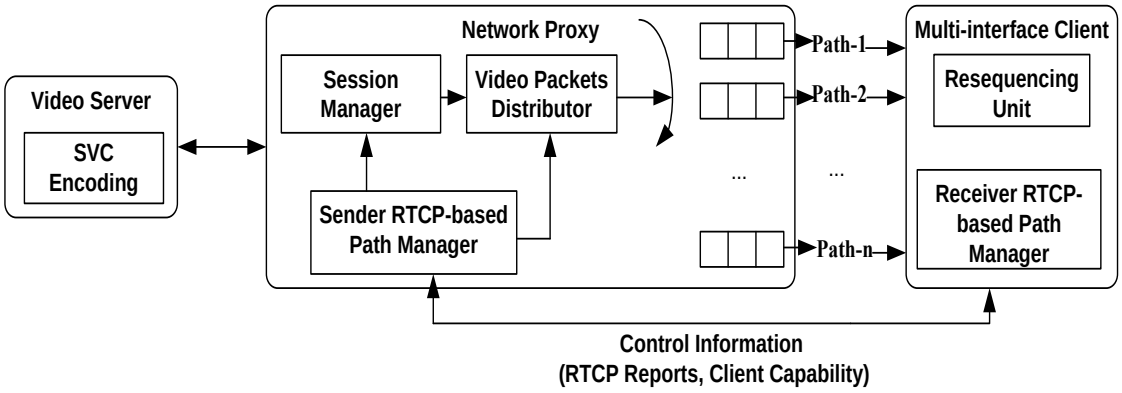

Fig. 5.4 Functional Components to Support Multipath Video Streaming Solution

5.2 The Proposed Video Packets Distributor

Previous work on multipath H.264 SVC video streaming considered the transmission of SVC bit-stream over multiple paths using video packet striping decisions derived as a function of available bandwidth and/or end-to-end delay. The overall objective of the previous research was to exploit concurrent multipath transmission to mainly meet the playback requirements of a video stream. The previous work, however, overlooked the loss characteristics of the network paths, thus risking the transmission of high priority video data, such as the base layer NALUs, over the network paths with high loss rate. When such NALUs are lost, the loss can propagate to dependent NALUs, thus resulting in considerable video distortion. They also considered concurrent multipath transmission of SVC encoded video at flow-level, which is not optimal for accomplishing balanced traffic load across the different network paths. Unbalanced traffic load over the different network paths can lead to poor utilization of the aggregated bandwidth capacity and reduced overall throughput [74].

This thesis proposes a new multipath video streaming solution, which includes a video packets distribution solution that considers not only bandwidth and delay performance

characteristics of the network paths but also network paths' loss rates in its operation. The objective of the proposed multipath SVC video packets distributor (VPD) is to prevent loss of the important parts of the bit-stream since losing them can cause errors that can propagate across several other NALUs, thereby yielding perceptible video artefacts, which may not satisfy the viewer. Furthermore, the VPD also aims at scheduling the NALUs such that they can meet their playback deadlines. The VDP uses packet-level splitting for improved load balancing.

The proposed VPD scheduling process begins by compiling a list of network paths that can be used concurrently to deliver video packets to the multi-homed user's device. This list of usable network paths is called the active set, which is denoted by A. The scheduling is done in rounds, where a scheduling round spans a series of NALUs forming a group of pictures (GoP). The active set must be updated before the beginning of the next scheduling round to remove stale network paths and add 'fresh' ones to assure proper adaptation. The network paths in A are ordered in ascending order of their estimated end-to-end loss rates. Consequently, A can be defined as: $A = \{L_i: i=1, 2, \dots, n\}$ (1), where L_1 is the network path with the least loss rate, while L_n is a network path with the highest loss rate; n is the cardinality of A.

The proposed video packets distribution method divides NALUs within a GoP into base layer and enhancement layer queues. Within each queue, the NALUs are prioritized based on decoding dependencies, where a NALU is assigned lower priority than the NALUs it depends on for decoding. The queue q_{i-1} (NALUs from the video layer l_{i-1}) is more important than the queue q_i (NALUs from higher video layer l_i). Let T_i be the playback delay required for NALUs in q_i, which can be estimated as a function of frame rate. The estimated delivery time of NALUs from q_i using a set of selected network paths S is denoted by t_i^S, and this is a function of the bandwidth pooled from the different network paths in S and the size of the NALUs in q_i.

The proposed VPD assigns the different queues to the network paths starting from the highest priority queue to the lowest priority one. When q_i is assigned to a network path in A, the network path must have the lowest loss rate to minimize possible video packet losses. Also, the network path must have enough bandwidth capacity to serve NALUs from the queue to meet their playback, that is, T_i must be less than or equal to t_i^S. If the required playback cannot be met with the bandwidth resources of the selected network path (or set of network paths) and there are still more network paths from A that can be used, the next network path with the lowest loss rate is

added to S to increase the amount of pooled bandwidth. If the pooled bandwidth is adequate to meet the playback requirements of NALUs in queue q_i, any surplus capacity is pooled for use in the scheduling of the NALUs from the next queue. This ensures that the network paths with the least loss are utilized to minimize video distortion and therefore improve the streaming quality. This method of NALUs distributions across the network paths does not restrict a flow of NALUs (video packets) to one particular network path, which is desirable for efficient load balancing.

In the event q_{i-1}, containing NALUs from the lower layer, could not be scheduled for transmission at all, the NALUs from q_i, which are NALUs from the higher layer, must be discarded since they cannot be decoded by the receiver without the reference NALUs from q_{i-1}. When the NALUs in q_i are to be scheduled for transmission but the available aggregated bandwidth resources are not sufficient to guarantee timely delivery for some of the NALUs, the VPD performs adaptation by scheduling only the NALUs that are likely to meet their playback and discarding the rest of the NAULs that cannot meet their playback deadlines. Thus, unnecessary waste of bandwidth resources that could be incurred by transmitting NALUs that will end up discarded by the receiver can effectively be avoided.

The NALUs from any queue are transmitted concurrently over the selected set of network paths as Multipath RTP payload over UDP. It should be noted that the multipath streaming mechanisms applied here can be adapted for transmission using TCP as the transport protocol. One of the daunting challenges of concurrent multipath transmission is packet reordering. Packet reordering of the packetized NALUs is resolved by using the scheduling technique proposed in chapter 4. In other words, the proposed multipath streaming solution comprises two scheduling levels. The first level involves mapping the NALUs from the different layers to a set of network paths according to priority of the layer and the loss characteristics of the network paths. The second level involves the scheduling of the NALUs (video packets) traversing different network paths to combat reordering at the receiver. It should be noted that packetization of the NALUs is done according to rfc3984 [93].

The proposed multipath streaming system assigns the video packets to the different network paths on a per GoP basis, thereby leading to a scheduling round spanning a GoP. At the end of the current scheduling round, the video packets distributor must reconfigure its scheduling strategy for the next GoP to ensure adaptation to any changes in the network paths' characteristics.

As mentioned earlier, the network paths' characteristics are gathered using the RTCP feedback channel between the network proxy and the multi-homed device. The reporting frequency for the implemented RTCP feedback system is on a per-GoP basis, which can be reconfigured depending on the rate of change of the network paths' conditions. The operation of the proposed video packets distributor is illustrated by Algorithm III. The next Section discusses a comprehensive performance evaluation of the presented concurrent multipath streaming of H.264 SVC encoded video for multi-homed user's device in a heterogeneous network environment.

Algorithm III Striping Video Traffic Across Multiple Network Paths

Require: Active set (*A*); base layer & enhancement layer queues

1: **for** *i* to number of queues **then**
2: **if** q_i is base layer queue **then**
3: select and add lowest loss rate path to *S* for q_i;
4: estimate t_i^S; //surplus capacity from q_{i-1} is part of *S*
5: **if** $T_i \leq t_i^S$ **then**
6: schedule q_i on *S*;
7: **else if** $T_i > t_i^S$ && there are more paths to add **then**
8: add next low loss path to *S* to aggregate capacity;
9: got to step 4;
10: **else if** $T_i > t_i^S$ && there are no more paths to add **then**
11: determine α_i; //Fraction of schedulable NALUs
12: schedule α_i on *S*;
13: discard the rest of the NALUs;
14: **end if**
15: **else**
16: **if** q_{i-1} scheduled **then**
17: got to step 3;
18: **else**
19: discard all the NALU from q_i;
16: **end if**
17: **end if**
18: **end for**

5.3 Performance Evaluation of the Proposed Multipath Streaming Framework

Several experiments have been conducted to evaluate the performance of the proposed multipath SVC streaming mechanism. The objective of the experiments is to ascertain the efficacy of the proposed solution under different network path conditions. The experiments are based on a novel evaluation framework, which comprises the ns-3 simulation environment, real video sequences and the JSVM reference software for encoding the input video sequences and decoding the output sequences. The network simulator, ns-3, is a discrete event simulator developed to simulate and emulate Internet systems. The details of the proposed evaluation framework framework are discussed in Section 5.3.1. Section 5.3.2 discusses the objective performance metrics used to evaluate the video quality delivered by the proposed multipath video streaming system. In Section 5.3.3, the two reference multipath streaming solutions that have been compared with the solution proposed in the thesis are described. Section 5.3.4 discusses the different bandwidth aggregation scenarios considered to evaluate the different multipath streaming schemes. The experimental results are presented and discussed in 5.3.5.

5.3.1 The Proposed Evaluation Framework

Figure 5.4 illustrates the components of the proposed performance evaluation framework. For simplicity, the network proxy and the video streaming server functions are encapsulated into a single entity only referred to as the video streaming server. The functions on the streaming server include the JSVM [94] encoder to compress the video sequences in the H.264 SVC format, a bitstream extractor to derive an appropriate bitstream for the requested video, and a traffic generator to generate the ns-3 compliant traffic from the extracted bitstream trace. The functional entities on the multi-homed client include a traffic trace parser to read the received ns-3 trace and generate an appropriate received bitstream trace, which is used by the bitstream extractor on the client terminal to get the received bitstream. The received bitstream is then decoded by the JSVM decoder implemented on the multi-homed client device. Although Fig. 5.5 shows only two network paths between the streaming server and the multi-homed device, the experiments conducted in the evaluation are based on a multipath network configuration consisting of three network paths in the ns-3 simulation environment, which natively supports nodes with multiple network interfaces.

Real video sequences from [95] are used in the evaluation. The well-known Foreman and Coastguard sequences have been used. These test sequences are of different genres, and they exhibit varying spatial and temporal complexities as depicted in Figure 5.6. Each video sequence is encoded into H.264 SVC bitstream using the JSVM-based encoder. Then, a suitable sub-stream is extracted to produce a video trace that is used to generate the video traffic that can be sent over multiple network paths in the ns-3 simulation environment. The evaluation uses one layer for spatial and quality domains, and focuses on video quality adaptation based on different temporal layers. Specifically, three temporal layers—namely, the base layer and two enhancement layers—are considered.

The ns-3 multipath network simulation environment comprises point-to-point link configurations in the core and three wireless technologies to connect the multi-homed device to the network. ns-3 allows for the mobile device to install multiple network interfaces, which makes setting up a multipath network relatively easy. The configured wireless networks are the WLAN, Wimax and LTE. The experimental process begins with the encoding of the video test sequences to generate H.264 bitstream. From the generated bitstream, a suitable sub-stream is extracted and the corresponding trace is produced. The resulting bitstream trace is parsed by the traffic generator to produce the traffic trace that is used to create packet payload that can be transmitted within the ns-3 environment. The created packets are striped over the three network paths (WLAN, WiMAX and LTE) according to the scheduling specifications in Algorithm III. The scheduling specifications have been developed in C++ and incorporated into the ns-3 environment as the functions of the streaming server node.

When packets arrive at the receiving multi-homed device, they are retrieved from the different network interfaces, recording the packets' attributes such as arrival time, size, and sequence numbers to generate the received trace. The received trace is then used to generate the bitstream trace that the JSVM bitstream extractor uses to generate the received H.264 bitstream. Finally, the received bitstream is decoded into a YUV sequence. The quality of the reconstructed YUV sequence is evaluated in terms of the objective metrics in Section 5.3.2. The MSU Video Quality Measurement Tool [96] was used to determine the PSNR and SSIM values from the reconstructed YUV sequences.

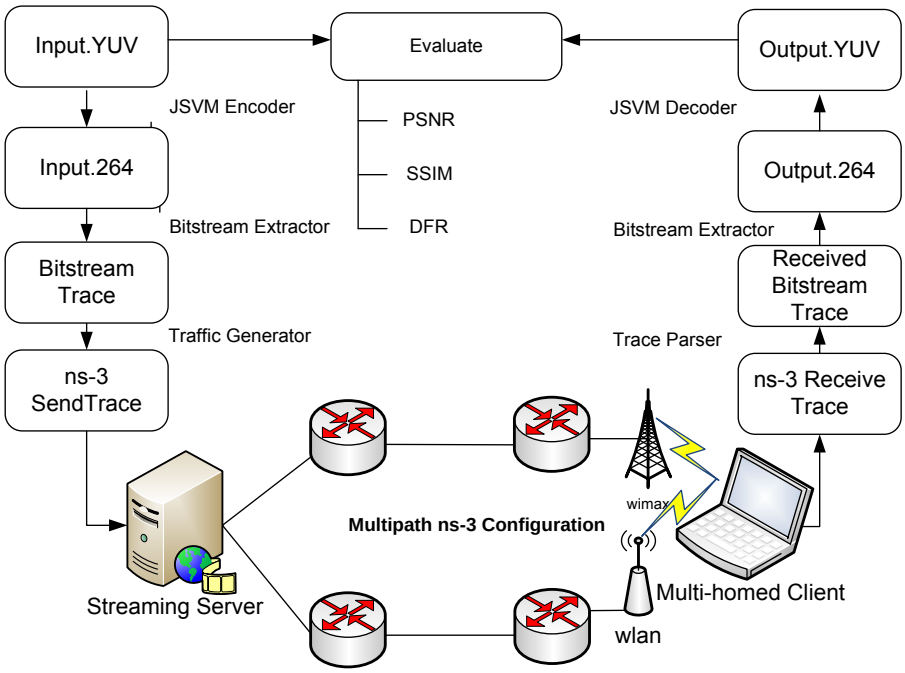

Fig. 5.5 The Proposed Evaluation Framework

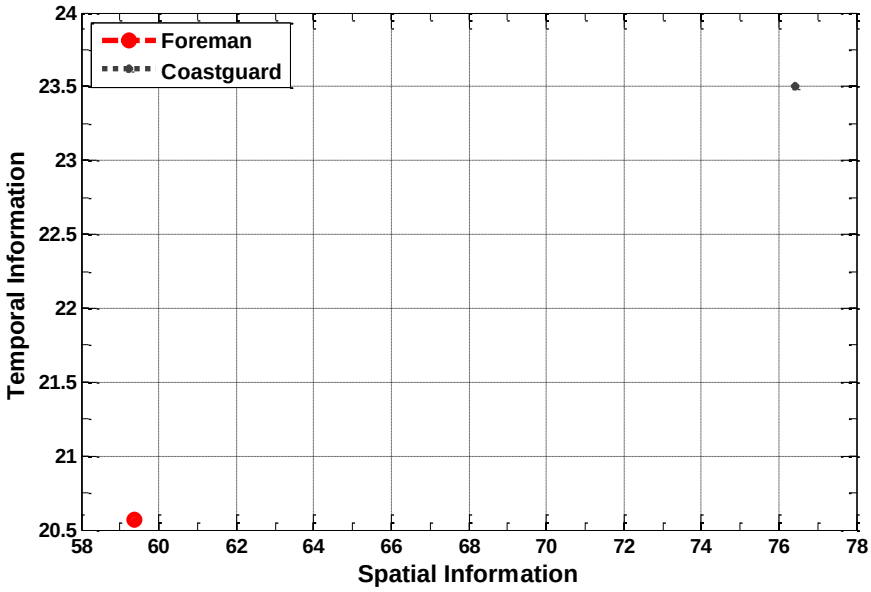

Fig. 5.6 Spatial and Temporal characteristics of the Foreman and Coastguard

(a) Foreman

(b) Coastguard

Fig. 5.7 Screenshots of the video test sequences

5.3.2 Objective Performance Metrics

The performance of the proposed concurrent multipath streaming scheme is evaluated in terms of peak-signal-to-noise ratio (PSNR), structural similarity index (SSIM) and decodable frame ratio (DFR). PSNR is an objective video quality metric, which captures the luminance and chrominance properties of an image [97]. PSNR is measured in decibels on a logarithmic scale using mean square error (MSE) between the reference and the received video sequences with the square of the highest sample value in a video image, which is 255 for 8-bit image. For an n-bit image, PSNR is determined as follows:

$$PSNR = 10\log_{10}\frac{(2^n - 1)^2}{MSE} \tag{2}$$

Mean Square Error and PSNR have gained popularity in the measurement of perceived video quality because of their ease of implementation. These are also well understood metrics mathematically. The human vision is more likely to notice the image artifacts in luminance. Therefore, the luminance component of PSNR is considered in this research.

PSNR relies on pixel-to-pixel comparison and may, therefore, not have accurate correction with the human observer. For this reason, the PSNR metric is used with the SSIM metric in the

thesis to accurately and fully capture the perceived video quality. SSIM is an objective video quality metric that measures structural similarities between two image sequences [97] in terms of luminance, contrast and spatial texture. Luminance is modelled as average pixel density, contrast is defined by the variance between the reference and the distorted video, and spatial texture is cross-correlation between the reference and the distorted videos. The motivation for SSIM is grounded on the notion that the human visual system assesses an image by mainly focusing on the image's structural properties. Finally, decodable frame ratio—measuring the percentage of decodable frames—is examined. The most preferable multipath video streaming model is one that can deliver a video with the highest values of PSNR and SSIM, and the lowest DFR under the considered experimental scenarios.

5.3.3 The Evaluated Multipath Video Streaming Solutions

In this research work, the proposed multipath streaming solution is compared with two representative multipath video streaming solutions. The first representative solution is called 'HighCapacity,' which represents the streaming solutions that work to optimize playback performance by switching transmission of the video packets between the network paths that offer the best performance in terms of bandwidth and/or delay. The multipath streaming solution in [47] exemplifies HighCapacity video streaming strategies. Secondly, the multipath video steaming proposal in [45], which is termed 'BestFit,' is considered. The BestFit streaming solution maps each SVC video layer to $\min(B_l) \geq b_i$, where B_l is available bandwidth on network path $l \in A(the\ set\ of\ active\ network\ paths)$ and b_i is the bitrate of the i^{th} SVC layer. The main objective of the BestFit is to improve video quality performance over multiple network paths while ensuring that the transmitted SVC video layers do not hog on bandwidth resources, thus assuring some level of fairness to other applications. The multipath video streaming system proposed in this thesis is simply called 'Proposed' in the presented evaluations and discussions. All in all, the evaluated multipath streaming solutions are *HighCapacity*, *BestFit* and *Proposed*. The experimental scenarios considered to evaluate these solutions are discussed in the next Section, and the results are discussed in Section 5.3.5.

Table 5.1 The investigated multipath video streaming solutions Features

Feature	Proposed Scheme	BestFit	HighCapacity
Bandwidth-aware	Yes	Yes	Yes
Loss-aware	Yes	No	No
Flow-level	Yes	Yes	Yes
Packet-level	Yes	No	No

5.3.4 Experimental Scenarios and Results

5.3.4.1 Scenario 1: Bandwidth Asymmetry with increasing loss rate disparity

In this part of experiment, a bandwidth aggregation scenario, where bandwidth from three network paths in a heterogeneous network are pooled is considered. Two of the network paths (path1 and path2) are configured to have identical available bandwidth capacities (800 Kbps), while the third network path (path3) is set to have higher capacity (1.6Mbps). The network paths' configuration in the ns-3 environment is such that the total delay on the network paths is dominated by the transmission delay, which depends on the network paths' bandwidth. The bandwidth settings are such that the pooled bandwidth capacity is enough to meet the target bitrates of the two test sequences. The packet loss rate is fixed at 0.01 for path1 and path2. On path3, packet loss rate is varied from 0.01 to 0.1. The purpose of the experiment under this test scenario is to determine the behavior of the investigated concurrent multipath video transmission mechanisms (5.3.3) under some amount of bandwidth asymmetry and increasing packet loss rate disparity on the network paths. The performance results are discussed below.

Table 5.2 Network Paths Characteristics for the two sequences: scenario 1

Network Path	Bandwidth (Kbps)	Loss Rate (%)
Path1	800	0.01
Path2	800	0.01
Path3	1600	0.01-0.1

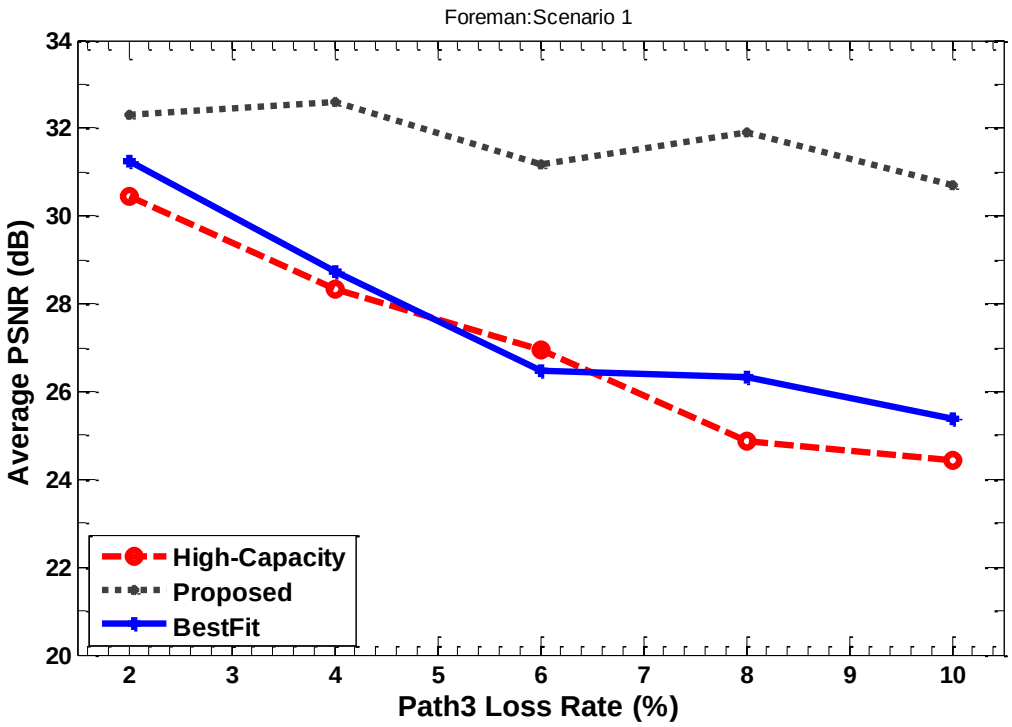

Fig. 5.8 PSNR Comparison for the Foreman Sequence under Scenario 1

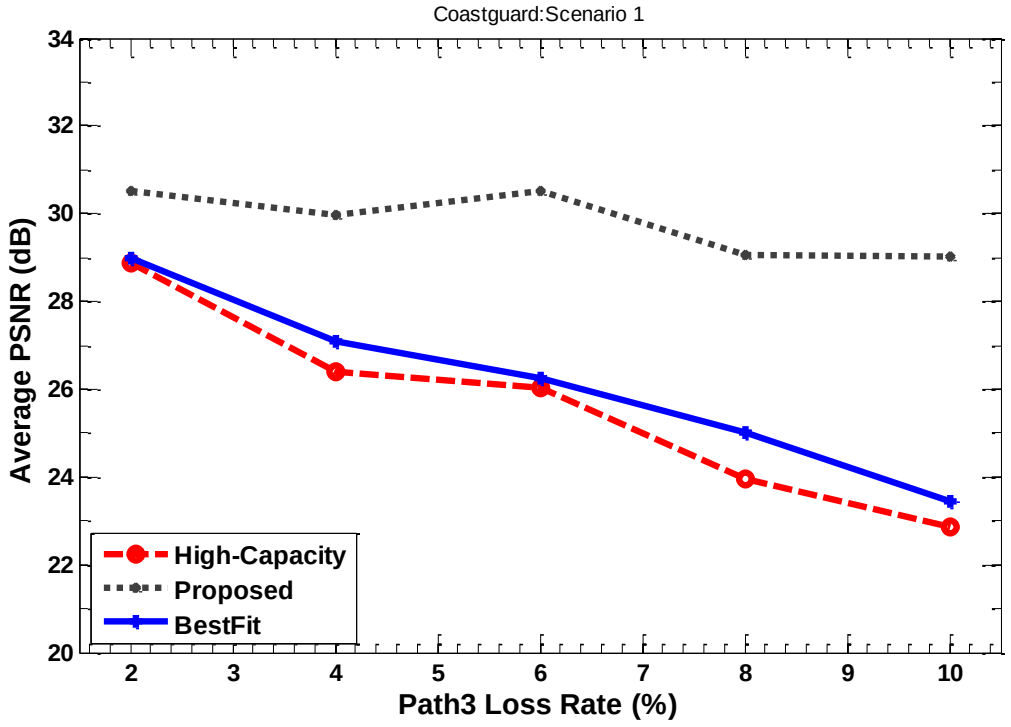

Fig. 5.9 PSNR Comparison for the Coastguard Sequence under Scenario 1

Fig. 5.8 and Fig. 5.9 show plots of the PSNR comparisons between the investigated multipath video streaming solutions for the foreman and the coastguard test sequences. It can be observed that the PSNR decreases with the increase in loss rate on path3 for all the investigated multipath streaming solutions. The PSNR performance under the proposed solution decreases more gradually, which leads to graceful video quality degradation. This is because the multipath streaming solution proposed in the thesis is aware of the network paths' loss rates and it tries not to schedule the reference video data (NALUs from reference frames) on the network paths that experience high loss rates. In the investigated bandwidth aggregation scenario, the proposed solution, whenever possible, does not schedule reference video data on path3, which helps to circumvent severe error propagation, which could occur due to decoding failure.

A sharp decrease in PSNR can be observed with the reference multipath streaming solutions (HighCapacity and BestFit), and that may lead to drastic video quality distortion, which may not be acceptable to the user. For HighCapacity solution, the poor PSNR performance can be attributed to the fact that the HighCapacity solution is oblivious to the network paths' loss rates as it only focuses on the highest bandwidth when selecting the network path to stream the video. In the investigated scenario, the network path with the highest bandwidth happens to also be encountering increasing loss rate, which results in the loss of a large number of video packets, including packets from the reference frames. The BestFit solution encounters a sharp decline in PSNR because it prefers the minimum bandwidth that satisfies the video layer's playback requirements, and if the network path providing the bandwidth has high loss rate such as path3, which serves one of the enhancement layers, a large number of packets are lost, thus resulting in poor PSNR performance. The BestFit performs better than the HighCapacity because some of the layers are served by the network paths with lower loss than path3, which is the only network path used by HighCapacity.

PSNR alone may not be the most accurate measure of the user's perceived video quality as it only makes pixel-by-pixel analysis, which is not consistent with the human visual system. To complement PSNR, SSIM is used to get a more complete objective video quality assessment. Fig. 10 and Fig. 11 plot the SSIM comparison for the two video sequences under the different multipath video streaming solutions. Similar to the PSNR performance, the proposed multipath video streaming solution delivers a video that is structurally more similar to the original sequences, thereby promising more acceptable video quality of experience to the user. It must be noted that

the PSNR and SSIM performance for the coastguard test sequence is a bit lower than for the foreman test sequence across all the investigated multipath streaming solutions. This can be attributed to the coastguard's higher bitrate and spatio-temporal complexity making it more prone to a higher packet loss whenever there are bit errors, thereby resulting in relatively lower PSNR and SSIM values.

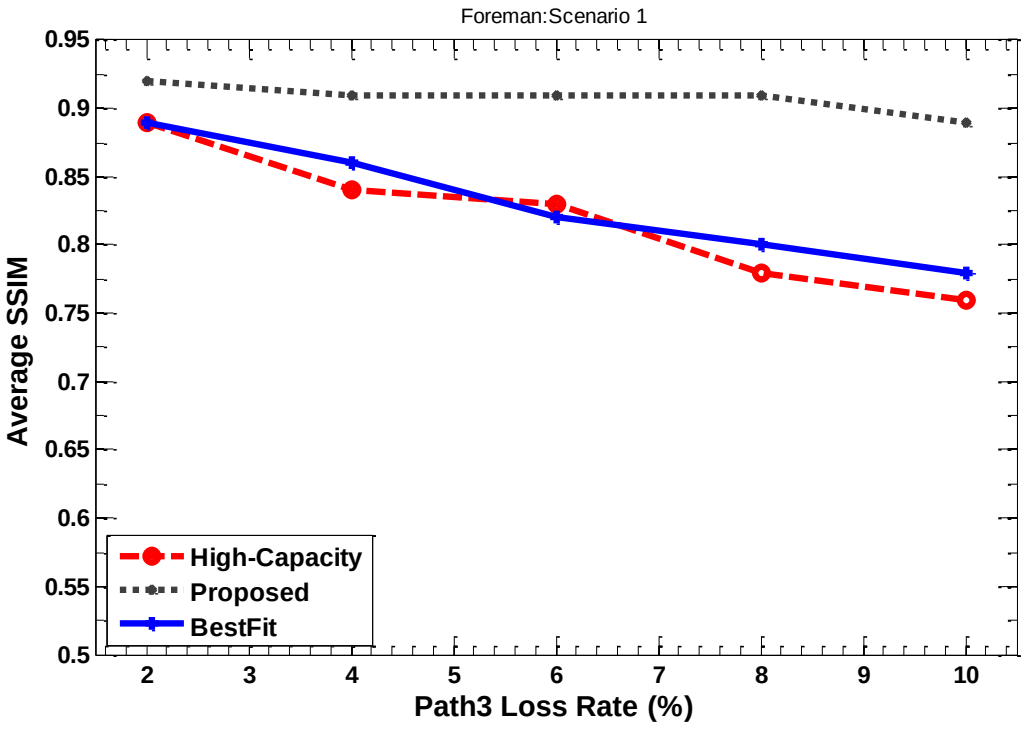

Fig. 5.10 SSIM Comparison for the Foreman Sequence under Scenario 1

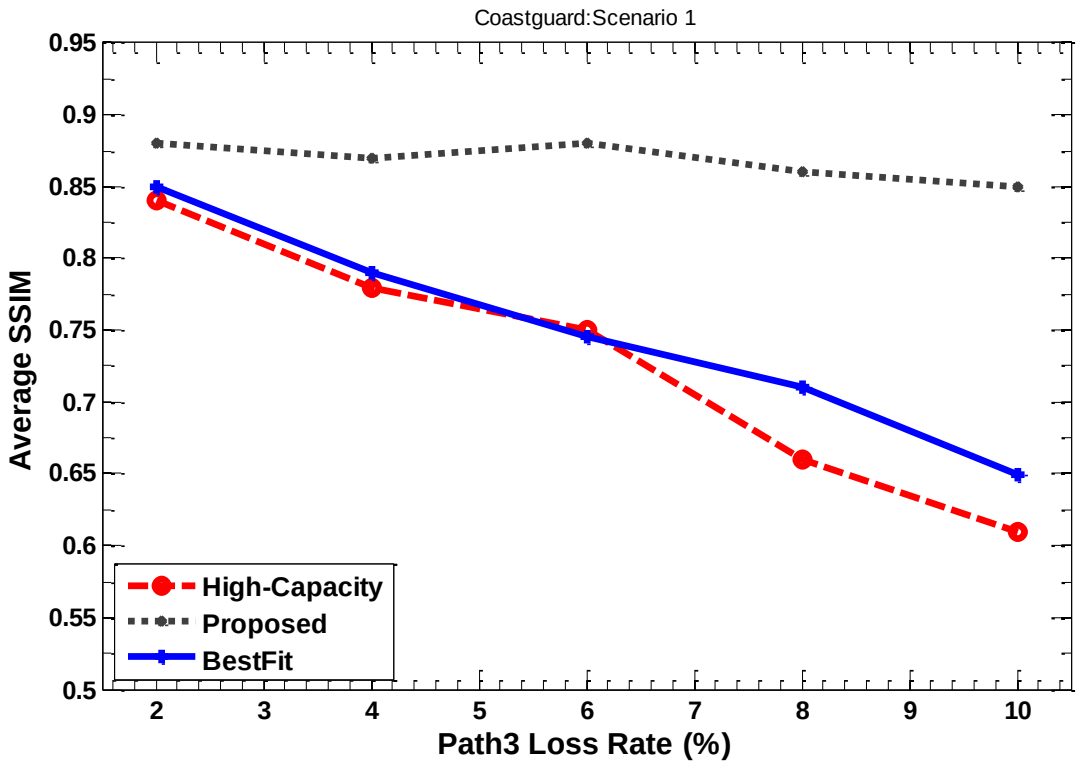

Fig. 5.11 SSIM Comparison for the Coastguard Sequence under Scenario 1

Fig. 12 and Fig. 13 depict plots of decodable frame ratio for the two test sequences under the investigated multipath video streaming solutions. As expected from the PSNR and SSIM results, the proposed multipath video streaming solution performs much better than the reference multipath streaming solutions in terms decodable frame ratio. That is, a larger number of received frames can be decoded by the multi-homed device under the streaming solution proposed in the thesis. This is because the proposed streaming solution avoids (whenever possible) scheduling reference NALUs on path3, which is the network path that is experiencing higher loss rates. Thus, a smaller number of reference video packets are lost, which results in a video that is mostly decodable, thus yielding more improved quality of experience. The two reference solutions, on the other hand, do not attempt to prevent the loss of reference frames by scheduling the frames away from path3 with the highest loss rate. Thus, they deliver a video stream with the least number of decodable frames, thereby affecting the received video quality negatively.

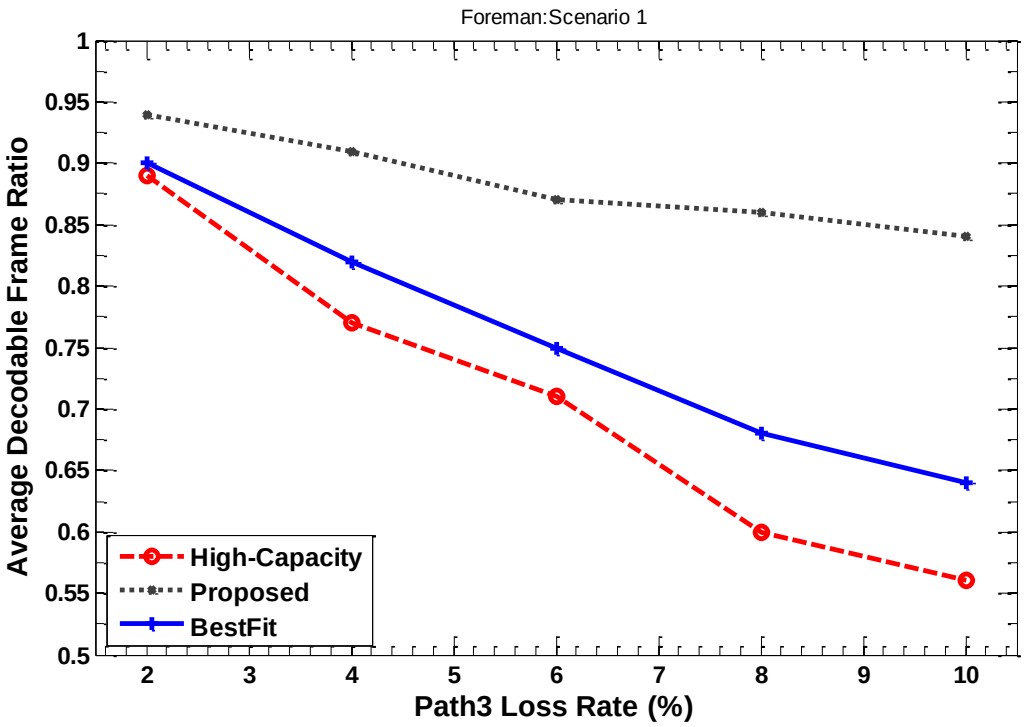

Fig. 5.12 Decodable Frame Ratio Comparison for the Foreman under Scenario 1

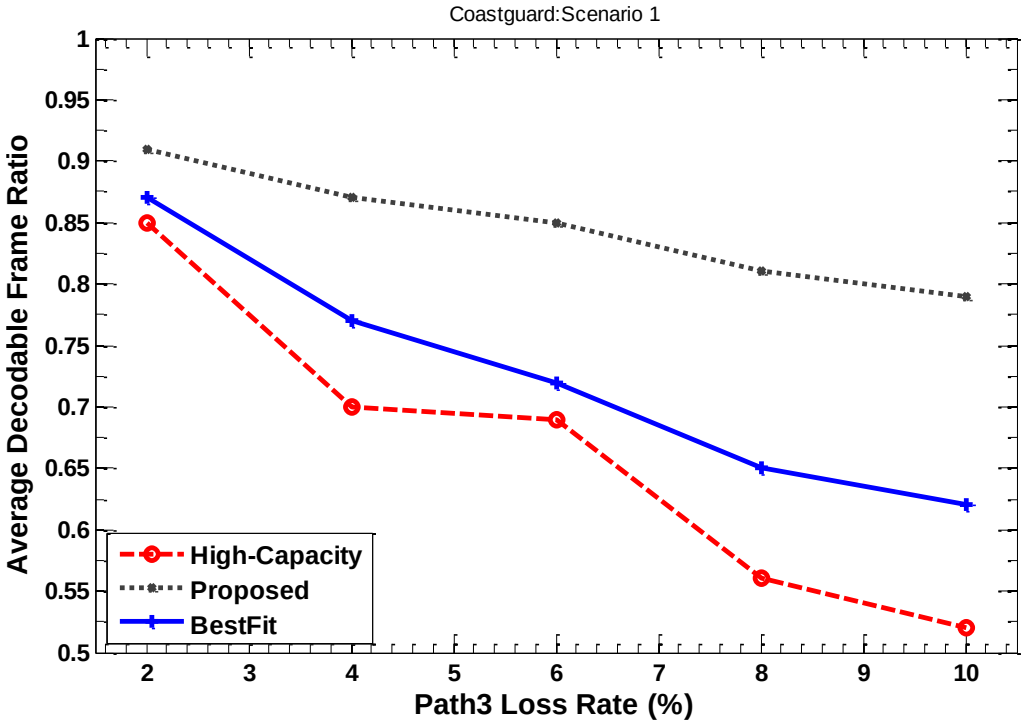

Fig. 5.13 Decodable Frame Ratio Comparison; the Coastguard under Scenario 1

5.3.4.2 Asymmetric losses with decreasing bandwidth capacities

In this bandwidth aggregation scenario, the three network paths have identical bandwidth capacities set at 600 Kbps for the Foreman sequence and 800 Kbps for the Coastguard sequence, ensuring that the pooled bandwidth can initially meet the sequences' target bitrates. Similar to scenario 1, the total delay on each network path is dominated by the network path's transmission delay. For each round of concurrent multipath video transmission, which spans a GoP, the bandwidth on each of the first two network paths (path1 and path2) is reduced by 10% until 50% of the original capacity is reached. This creates some bandwidth asymmetry with diminishing pooled bandwidth as a result of the bandwidth reductions on path1 and path2. The loss rates on the network paths are set as follows: 0.01 for path1, 0.05 for path2 and 0.1 for path3.

The purpose of this part of the experiment is to observe how the different concurrent multipath video streaming mechanisms can adapt and achieve graceful video quality degradation in the face of diminishing bandwidth capacity and heterogeneous loss rates across the different network paths. The simulated bandwidth aggregation scenario can be encountered in a real heterogeneous wireless network environment, where bandwidth reduction may occur as a result of an increase in competing traffic, user mobility resulting in less favorable coverage, and channel errors that often plague wireless transmission.

An efficient concurrent multipath streaming solution should be able to adapt the concurrent multipath video transmission to the reduction in pooled bandwidth to avoid drastic deterioration of the video quality. While adapting the video load to the changes in bandwidth, it is desirable for an efficient concurrent multipath streaming scheme to be aware of the disparity in loss rate across the different network paths so that the most important parts of the video stream, such as the NALUs from the base layer, may not be transmitted (whenever possible) via the network paths with the highest loss rate.

Table 5.3 Network Paths Characteristics for Foreman: scenario 2

Network Path	Bandwidth (Kbps)	Loss Rate (%)
Path1	600-300	0.01
Path2	600-300	0.05
Path3	600	0.1

Table 5.4 Network Paths Characteristics for Coastguard: scenario2

Network Path	Bandwidth (Kbps)	Loss Rate (%)
Path1	800-400	0.01
Path2	800-400	0.05
Path3	800	0.1

Fig. 14 and Fig. 15 illustrate the PSNR results for the Foreman and the Coastguard sequences under the three investigated concurrent multipath video streaming solutions. It can be observed that the proposed concurrent multipath video streaming solution generally achieves higher PSNR values than the reference solutions (BestFit and HighCapacity). This can be attributed to the proposed concurrent multipath video streaming solution's ability to gracefully scale down the transmitted video as the network bandwidth conditions deteriorate as it is the case in the investigated scenario.

When bandwidth is reduced on some of the network paths, resulting in the pooled bandwidth that is below the target video bitrate, the proposed multipath video streaming solution only maximizes the transmission of the NALUs that constitute the reference frames, discarding the low priority NALUs that cannot fit within the available bandwidth. Furthermore, the proposed solution attempts to sway the transmission of reference NALUs away from the network paths that are experiencing high loss rates whenever possible, thus avoiding drifting errors. In the investigated scenario, the proposed video streaming solution can only transmit the base layer NALUs via path3 when pooled bandwidth on the other two network paths cannot meet the base layer's bitrate. Otherwise, the base layer is always striped across path1 and path2, thus avoiding high packet losses on path3.

The BestFit achieves poorer PSNR performance than the proposed video streaming solution because of two main reasons. First, when bandwidth on path1 and path2 has been reduced to a point that the base layer's target bitrate can no longer be met, path3—with the highest bandwidth and loss rate—is used for the base layer transmission, thus resulting in higher packet losses, which significantly degrade the perceived video quality. Second, after scheduling the base layer for transmission, say on path3, if the available bandwidth on any of the other network paths is not sufficient to meet the enhancement layer target bitrate, the enhancement layer NALUs are

not transmitted at all even if the pooled bandwidth would meet the target bitrate. This results in only the basic quality level being achieved at best. The BestFit solution performs bandwidth aggregation at flow-level, which poorly exploited pooled bandwidth from the different network paths. The proposed solution uses both flow-level and packet-level splitting, allowing NALUs from a video layer to be transmitted on one network path and the NALUs to be striped across multiple network paths in the event a single network path does not have adequate bandwidth. The proposed video streaming solution can better utilize concurrent multipath transmission, which yields superior multipath video streaming performance.

The proposed multipath video streaming solution outperforms the HighCapacity scheme because HighCapacity can only use one network path at a time, switching to a higher bandwidth network path whenever possible. In the investigated scenario, the HighCapacity solution uses path3 with the highest bandwidth. However, path3 also happens to experience the highest loss rate, which results in high packet losses, and thereby yielding much lower PSNR values. Also, the one network path that HighCapacity uses may not individually meet the target bitrate for the video and this is true for the Coastguard sequence, which has been encoded at a higher bitrate than the amount of bandwidth available on path3. Because of this, HighCapacity fails to meet the some of the video packets' playback, resulting in the packets being discarded, which leads to poor video quality.

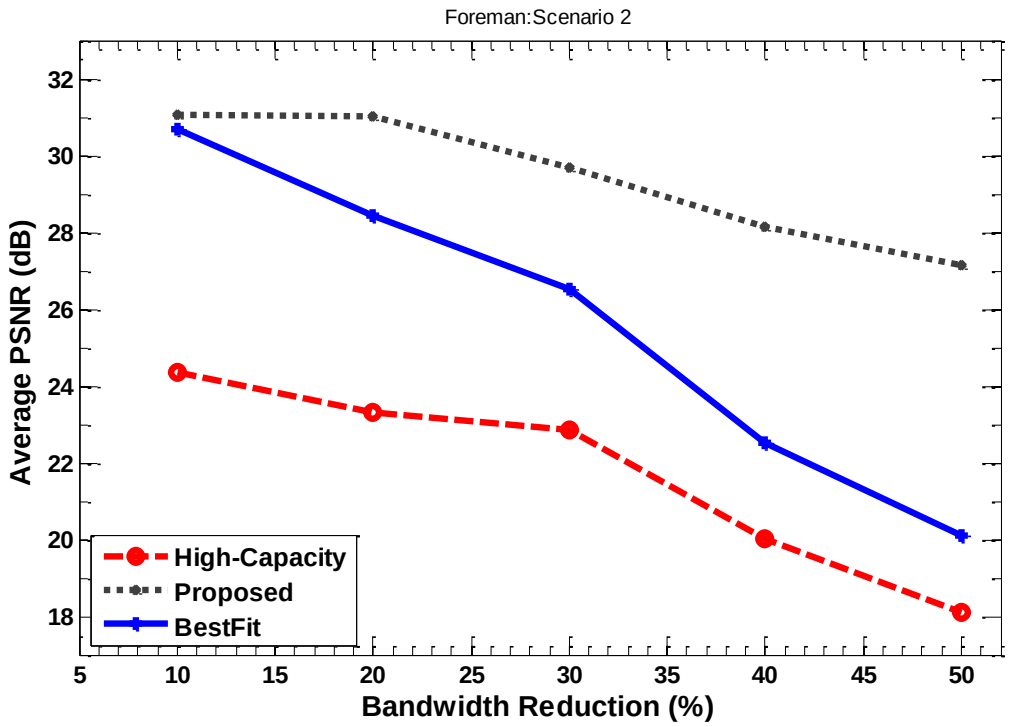

Fig. 5.14 PSNR Comparison for the Foreman Sequence under Scenario 2

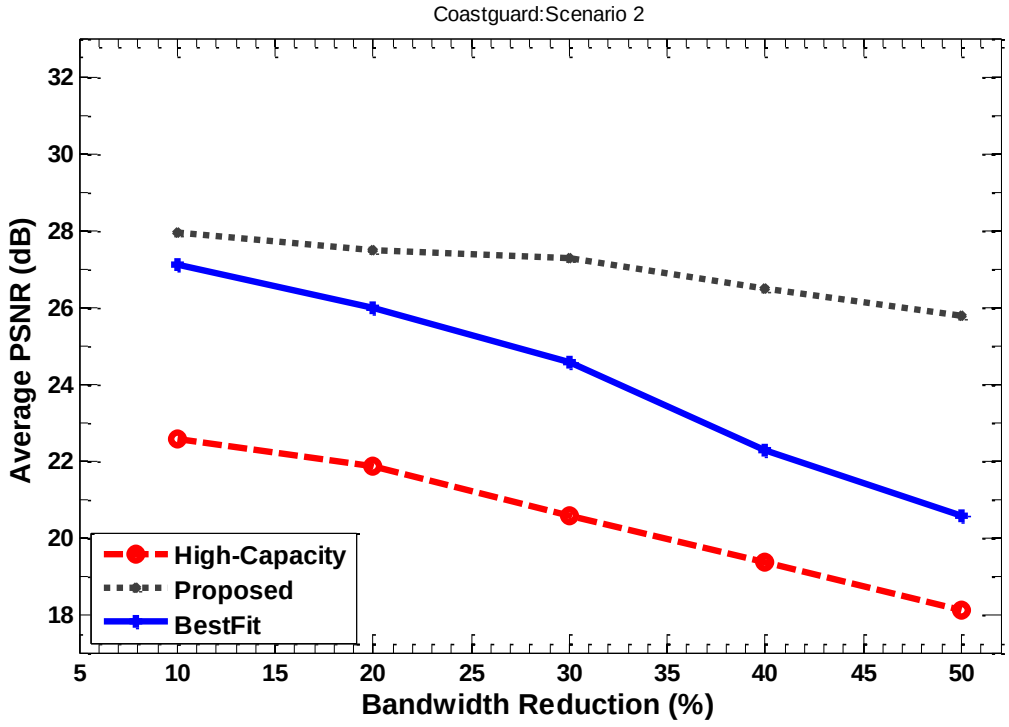

Fig. 5.15 PSNR Comparison for the Coastguard Sequence under Scenario 2

Fig. 5.16 and Fig. 5.17 illustrate the SSIM comparisons for both the Foreman and Coastguard sequences under the three investigated multipath video streaming solutions. The proposed multipath streaming solution outperforms the two reference schemes in terms of SSIM. That is, the proposed solution delivers the video that is more similar to the undistorted video, which shows that the solution can adapt well to diminishing bandwidth and the heterogeneous loss rates across the different network paths, thus enabling graceful degradation of the video streaming quality over multiple networks paths.

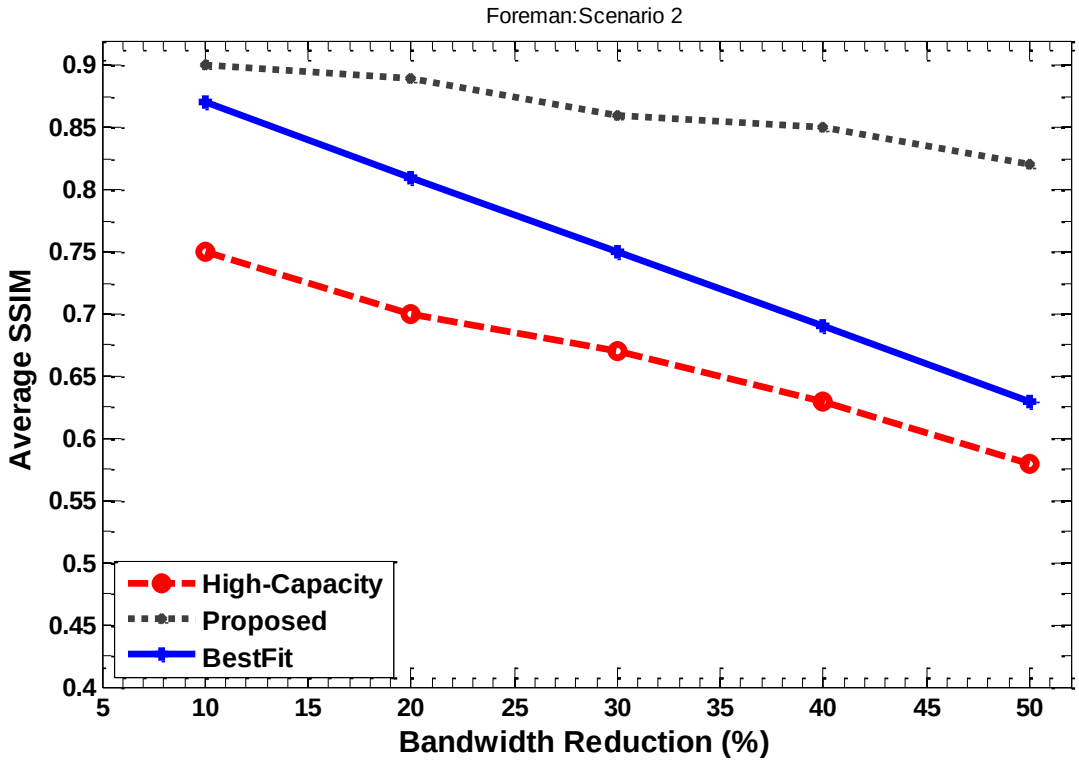

Fig. 5.16 SSIM Comparison for the Foreman Sequence under Scenario 2

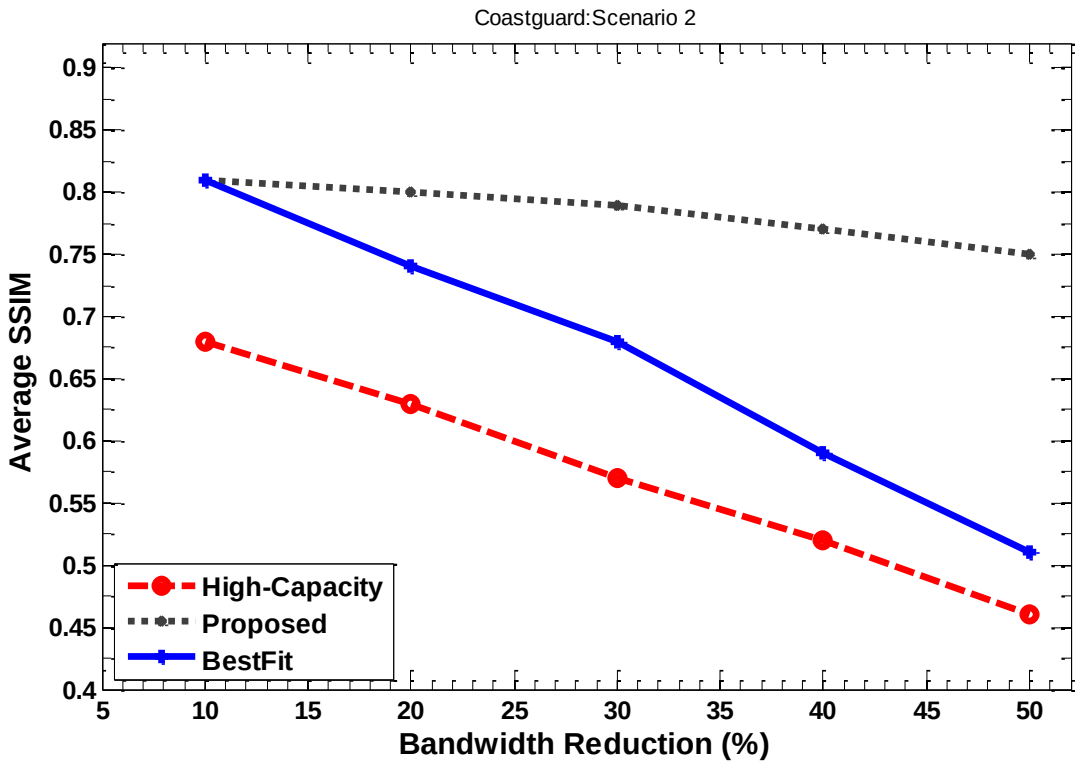

Fig. 5.17 SSIM Comparison for the Coastguard Sequence under Scenario 2

The decodable frame ratio (DFR) results for the two test sequences are shown in Fig. 18 and Fig. 19. As expected, the proposed concurrent multipath video streaming solution performs better than the reference multipath video streaming schemes in terms of DFR. The reason is that—as bandwidth capacity shrinks on the different network paths—the proposed scheme transmits mostly the NALUs that come from reference frames and it also attempts to transmit these reference NALUs via the network paths with the least loss. Thus, the proposed multipath video streaming solution experiences low loss of reference video packets, which improves the decodability of dependent video packets.

The reference schemes, on the other hand, do not offer the same level of protection of reference NALUs as they are oblivious to the network paths' loss characteristics. They also do not prioritize transmitting the reference video packets before the dependent ones, which leads to loss of the reference video packets in the event the reference video packets do not meet their decoding deadlines. When reference video packets are transmitted via network paths' with high packet loss rates and the packets are also not prioritized for transmission before the dependent video packets,

the video packets are inevitably lost and the decodability of the received video stream is negatively affected. This is evidenced by the poor decodable frame ratio results of the BestFit and HighCapacity schemes for both the Foreman and the Coastguard sequences. The BestFit outperforms HighCapacity because, to some extent, it is able to leverage aggregated bandwidth capacity.

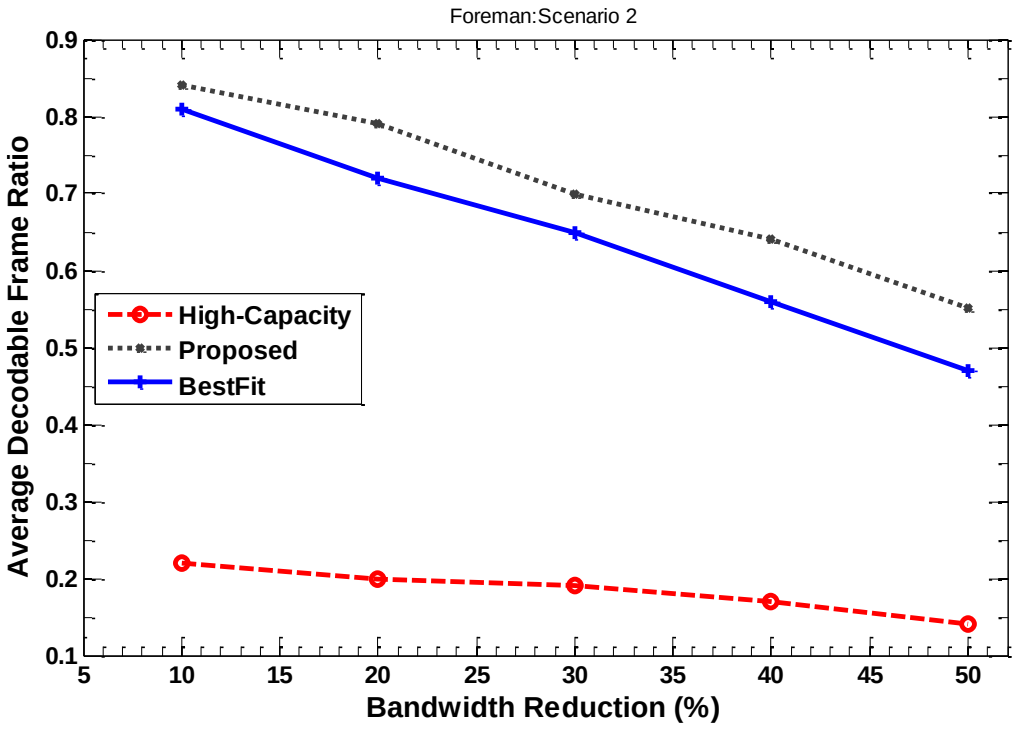

Fig. 5.18 DFR Comparison for the Foreman Sequence under Scenario 2

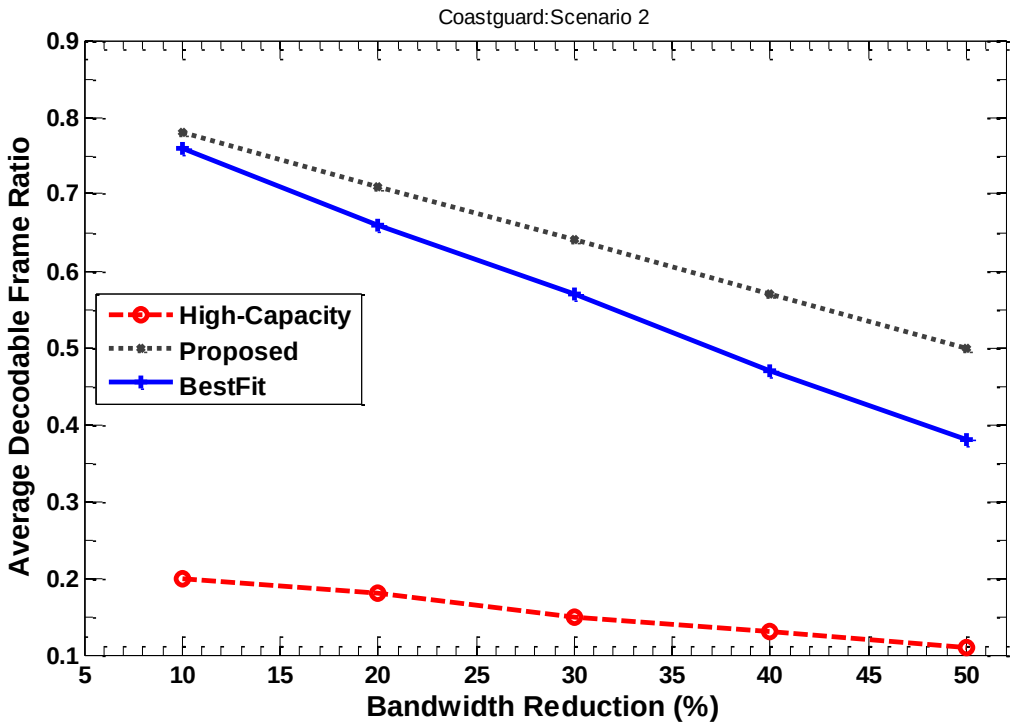

Fig. 5.19 DFR Comparison for the Coastguard Sequence under Scenario 2

5.3.4.3 Scenario 3: Dynamic Variation in Bandwidth and Packet Loss Rate disparity

In this bandwidth aggregation scenario, bandwidth and packet loss rate on each of the network paths vary dynamically. This is accomplished by configuring each network path to randomly take bandwidth values between 50% and 100% of the total bandwidth required to meet the target bitrate of each of the video test sequences. The loss rate on each network path varies randomly between 0.01 and 0.2. The interval for the variations is set to the length of a GoP. In other words, the network paths randomly take new values of bandwidth and loss rate for every GoP transmission. The purpose of the experiment is to demonstrate the ability of the investigated multipath video streaming solutions to adapt the video traffic load to dynamic changes in the network paths' characteristics in order to improve and/or maintain good video streaming quality.

Table 5.5 and Table 5.6 show average PSNR, SSIM and DFR results for the Foreman and the Coastguard sequences, respectively. The proposed multipath video streaming solution performs better than the two reference multipath streaming solutions in terms of PSNR, SSIM and DFR. This is because the proposed scheme can adapt to dynamic variations in bandwidth and loss

by appropriately reconfiguring the load across the network paths and finding reliable network paths to stripe reference video packets. The BestFit performs worse than the proposed multipath streaming because it is oblivious to the network paths' loss characteristics, and in the event the best-fitting network path happens to experience high loss rate, a large number of video packets are lost. The loss becomes even more devastating if the lost video packets are reference video packets. The loss of reference video packets propagates to several dependent packets, which often results in significant video distortion.

HighCapacity offers the worst performance in terms of average PSNR, SSIM and DFR because of two main reasons. First, HighCapacity uses only one network path at a time, switching to the next better performing network path whenever possible. However, the available network paths individually may not always have enough bandwidth to meet the target bitrate, thereby resulting in some of the video packets being discarded due to failure to meet playback deadline. When the discarded packets happen to be the reference video packets, the video quality degrades significantly due to decoding errors that propagate through a group of pictures. Second, HighCapacity incurs further losses in the event the HighCapacity network path has high loss rate, which exacerbates loss caused by limited bandwidth. Even though BestFit performs worse than the proposed solution, its performance is better than that of HighCapacity because it's able to deliver higher throughput as a result of its ability to utilize multiple network paths simultaneously. Another reason for HighCapacity's suboptimal performance may be attributed to network path switching, which occurs whenever there is a change in the network paths' bandwidth, which renders a network path different from the current as the highest capacity network path. Network path switching can introduce significant delays that can lead to video packets missing their playback deadline, thus causing video distortion.

Table 5.5 Foreman results under dynamic network path conditions

Solution	Average PSNR	Average SSIM	Average DFR
Proposed	32	0.85	0.9
BestFit	26	0.6	0.6
HighCapacity	22	0.5	0.4

Table 5.6 Coastguard results under dynamic network path conditions

Solution	Average PSNR	Average SSIM	Average DFR
Proposed	30	0.85	0.8
BestFit	23	0.5	0.4
HighCapacity	19	0.4	0.3

5.4 Chapter Summary

The chapter presented the details of a new concurrent multipath video streaming solution to improve quality of H.264 SVC video-on-demand service for users with multi-homed devices. The solution stripes video packets from an H.264/SVC encoded video stream across multiple network paths for concurrent transmission in a heterogeneous network. The goal is to adaptively deliver the video packets to a multi-homed device to meet their playback deadline and to prevent loss of reference video packets to avoid frame error propagation, which often degrades the received video quality. An evaluation framework has been developed and implemented in the ns-3 environment to evaluate the efficacy of the proposed multipath video streaming solution under various network path characteristics. The experimental results have shown that the proposed multipath video streaming solution can accomplish better video quality performance in terms of objective video quality metrics such as PSNR, SSIM and DFR.

An interesting direction for future work is to evaluate the proposed multipath video streaming solution in a suitable testbed environment to get a much closer to real life efficacy. Also, evaluating the scheme for interactive video is another area of research that could be pursued in future.

Chapter 6

Concurrent Multipath Transmission with Multipath TCP

Transmission Control Protocol (TCP) is the most widely used transport layer protocol. TCP has gained popularity due to its connection-oriented mechanism, which guarantees reliable data delivery. TCP, however, is not suited for the emerging concurrent multipath transmission in heterogeneous networks. In concurrent multipath transmission, multiple network paths are pooled into a single logical path, thus aggregating bandwidth to improve performance of demanding TCP applications. Concurrent multipath transmission can cause high data reordering. TCP can only deal with data reordering up to three displacements. Beyond that, spurious retransmissions are triggered and the congestion window is aggressively throttled, which significantly degrades the TCP throughput performance.

The IETF has developed a multipath variant of TCP, which is called Multipath TCP (MPTCP), to improve data delivery performance using multiple heterogeneous network paths simultaneously. At the core of MPTCP's concurrent multipath transmission are congestion control and data distribution mechanisms. The focus of this thesis is on the data distribution mechanisms (schedulers), which are responsible for striping data segments from the same application across several concurrent network paths. The network paths in a concurrent multipath transmission system are often heterogeneous, thus calling for a scheduler that can adapt to the heterogeneity. In this thesis, an adaptive MPTCP scheduler has been proposed. The performance of the proposed scheduler has been evaluated and compared with two reference MPTCP schedulers. The results have shown that the proposed scheduling mechanism adapts better to heterogeneous network path conditions and therefore achieves higher MPTCP throughput than the default schedulers.

The rest of the chapter is organized as follows. Section 6.1 presents an overview of MPTCP, discussing the goals of MPTCP and the functional components that constitute MPTCP. Section 6.2 discusses the design of the proposed MPTCP scheduler. In Section 6.3, the experimental evaluation results are presented and the performance of the proposed MPTCP scheduler is critically discussed. Section 6.4 concludes the chapter and proposes future work.

6.1 Overview of Multipath TCP

The IETF has defined Multipath TCP in RFC6824 as an extension to TCP [31]. The extension adds multi-homing and concurrent multipath transport features to TCP. The motivation for the extension is to leverage multiple network paths in data centres and exploit the multiple network interfaces that come embedded in a plethora of modern communication devices. With concurrent multipath transmission enabled for hosts with multiple network interfaces, demanding applications, such as high definition television, can adequately be supported, thus improving the user's quality of experience. The following subsections present the goals and features of MPTCP.

6.1.1 Goals of Multipath TCP

Multipath TCP is a modified version of the traditional TCP to enable concurrent multipath transmission between multi-homed hosts. The primary goal of MPTCP is to pool resources from several network paths to scale up capacity for high bandwidth demands from emerging high bandwidth applications, such as high definition video streaming and online gaming. The pooled network paths can be used by an application as a single logical path that has larger capacity to improve throughput performance. The key performance drivers for the development of MPTCP are as follows:

- To improve resilience by allowing a multi-homed host to maintain multiple network paths, routing application packets through one network path at a time and switching to any other available network path in case of failure or low performance on the primary path.
- To use multiple network paths simultaneously for a single packet flow, where the packets are striped across the network paths for parallel transmission. This is often referred to as concurrent multipath transmission.

The design of Multipath TCP also makes it possible for a multi-homed MPTCP host to establish communication with a host that only supports the traditional TCP. That is, MPTCP has the ability to fall back to single path TCP mode in cases where it is not supported. Besides throughput enhancement goal, MPTCP has been designed to enable loading balancing among several network paths. When congestion window on one network path fills up, traffic can be moved to other network paths that have larger available space in their respective congestion windows.

6.1.2 Features of Multipath TCP

Multipath TCP achieves its goals by splitting the transport layer into two functional components as illustrated in Fig. 6.2. The first component is the MPTCP layer, which fulfils the transport protocol's end-to-end operation. Below the MPTCP layer is a layer that is made of several sub-flows, each of which represents an independent TCP flow. The different TCP sub-flows make MPTCP appear as addressable standard TCP sessions to network elements, especially middle boxes, thus ensuring network compatibility. Each MPTCP sub-flow is typically established on its own network path, which may be disjoint from the others. The MPTCP layer also coordinates the operation of the different sub-flows, effectively making the application unaware of the underlying sub-flows. To this end, MPTCP implements the following functions.

a) *Path management*: This is responsible for network path discovery and connectivity checks between the end hosts. Path management can also include suitable mechanisms to collect network path characteristics such as delay, segment loss and throughput. These characteristics may be used to decide how to appropriately stripe segments across the network paths.

b) *Packet Scheduling*: A scheduling function stripes data segments from the MPTCP layer across the available TCP sub-flows. The scheduler can use one sub-flow at a time, switching to the best performing sub-flow any number of times. It can also stripe the segments simultaneously across multiple sub-flows for parallel transmission (concurrent multipath transmission). This thesis studies scheduling for concurrent multipath transmission. Most implementations of MPTCP use simple round robin and/or low round trip time (LowRTT) scheduling [98]. The problems with round robin are poor load balancing and high packet reordering. Round robin serves different sub-flows equally, which leads to load imbalance in the event the sub-flows do not have equal capacity. Round robin's lack of adaptation to network path characteristics, as it has been discussed in Chapter 4, yields high reorder entropy, which is not desirable for optimal TCP performance. LowRTT, on the other hand, adapts to path characteristics in order to optimize end-to-end delay and throughput performance by preferring sub-flows with the quickest responsive time (shortest RTT). However, LowRTT does not solve the problem of segment reordering, which often plagues concurrent multipath transmission. LowRTT assigns segments to the lowest RTT sub-flow until its congestion window is full and then moves on to the next lowest RTT sub-flow. RTT does not accurately predict the expected arrival times of

the different segments across the different sub-flows, which puts the LowRTT MPTCP scheduler at a risk of segment reordering. When segment reordering occurs, it can degrade MPTCP throughput performance significantly. The proposed MPTCP scheduler attempts to solve the problem of segment reordering by ensuring a scheduled segment has a chance of arriving at the receiver before segments scheduled after it. This is accomplished by scheduling a segment on a sub-flow with the shortest expected arrival time. The scheduler further prioritizes retransmitted segments to solve possible receiver window blocking. The details of the proposed MPTCP scheduler are discussed in Section 6.2.

c) *MPTCP Sequence Numbering*: MPTCP uses two levels of sequence numbering. The first level marks data segments at the MPTCP layer. The sequence numbers at this layer are called data sequence numbers (DSNs). The DSNs help to keep track of the overall sequence of data segments belonging to a particular application. DSNs are critical for resolving reordering at the MPTCP receiver. The second level of sequence numbering applies at the sub-flow level, where each sub-flow independently marks the data segments assigned to it by the scheduler. Fig. 6.1. illustrates MPTCP sequence numbering.

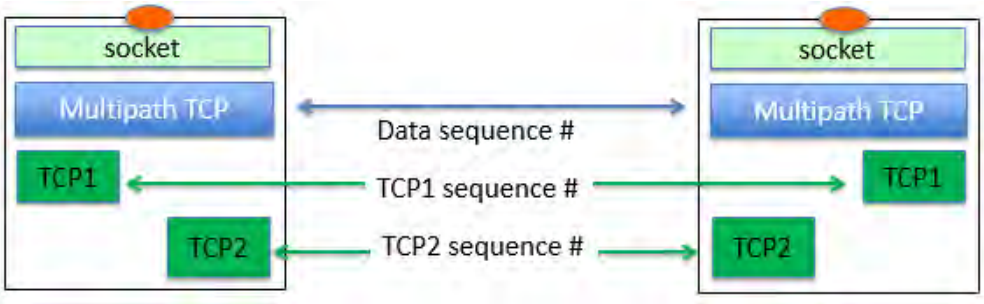

Fig. 6.1 MPTCP Sequence Numbering

d) *Congestion Control*: This function helps to control traffic flowing to the MPTCP receiver and ensures that congestion can be curbed to avoid loss and accomplish improved throughput performance. Several congestion control mechanisms have been proposed for MPTCP. The notable ones include linked increase, opportunistic linked increase, balanced increase and wVegas Delay based congestion [99]. Congestion control is also important for ensuring that MPTCP connections do not hog on aggregated bandwidth resources, thus allowing traditional TCP connections to attain reasonable throughput performance.

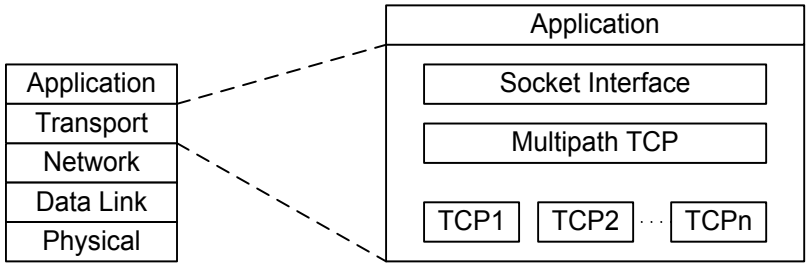

Fig. 6.2 The Network Protocol Stack with Multipath TCP

6.2 The Proposed MPTCP Segment Scheduler

This section presents the details of the proposed MPTCP scheduler. The proposed MPTCP scheduler makes scheduling decisions based on estimated expected arrival time, where the shortest time is preferred for a segment. Moreover, the scheduler includes a simple mechanism to schedule retransmitted data segments in a way that can improve their chance of making it to the MPTCP receiver. It is necessary to maximize successful transmission of retransmitted segments to avoid prolonged receiver buffer blocking. At the MPTCP receiver, a segment re-sequencing unit is implemented to complement the proposed MPTCP scheduler. The re-sequencing unit orders arriving segments and then sends them to higher layers. The re-sequencing unit also includes a mechanism to detect losses early enough to enable retransmission of the lost segment before a retransmission timer timeout.

The proposed MPTCP scheduler predicts the segments' expected arrival sequence at the MPTCP receiver based on estimated expected arrival time on each sub-flow in the active set (A). The active set is formed by sub-flows that have free space in their congestion windows. The active set must be periodically refreshed, adding new sub-flows and removing the stale ones (sub-flows that have failed and those that have full congestion windows). To en-queue a ready-to-send segment in the transmission buffers of the different sub-flows in the active set, the expected arrival time of the segment is estimated for each sub-flow as follows. Let T_i^l be the expected i^{th} segment's time on sub-flow l, after which the segment arrives at the receiver. Assuming there are already k segments waiting for service ahead of the i^{th} segment in the transmission buffer of sub-flow l, T_i^l can be estimated according to equation (1).

$$T_i^l = q_i^l + t_i^l \quad (1)$$

In equation (1), q_i^l is the queuing delay for segment *i*, which—in fact—is the time required to de-queue the k segments that are already waiting ahead of segment i in the sub-flow's transmission buffer. Let the size of the k segments in the buffer be denoted by S_k^l. Assume a sub-flow clears the segments from the buffer at the rate that is equal to its bandwidth capacity. Let the bandwidth capacity of sub-low l be denoted by B_l. q_i^l is, therefore, given by equation (2).

$$q_i^l = \frac{S_k^l}{B_l} + \gamma \quad (2)$$

In the equation, γ is the residual transmission time of the segment that has been de-queued but has not yet arrived at the receiver. γ is simply the difference between the segment's estimated arrival time and elapsed time since departure from the transmission buffer. t_i^l is the estimated delivery time for segment *i* after being de-queued, and this time is a function of the segment's size (S_i), B_l, and (P_l) propagation delay on the sub-flow l as shown in equation (3). To get the bandwidth estimate for the sub-flow, the packet pair technique in [100] can be used.

$$t_i^l = \frac{S_i}{B_l} + P_l \quad (3)$$

After estimating T_i^l on all the sub-flows in the active set, the proposed MPTCP scheduler selects the sub-flow that offers $\min(T_i^l)$ to deliver the segment to the receiver. When several sub-flows offer equal $\min(T_i^l)$, the scheduler will select the sub-flow that has the largest free space in the congestion window. This helps to spread the traffic load across lightly loaded sub-flows to avoid congestion, thus accomplishing the load balancing goal of MPTCP. When the estimated arrival time and congestion window across the different sub-flows are equal, the proposed MPTCP scheduler may further be optimized by selecting the sub-flow with the lowest loss rate.

The sub-flow selection strategy under the proposed MPTCP scheduler can be expressed as in equation (4). *Sub-flow** denotes the selected sub-flow.

$$subflow^* = \begin{cases} l, & T_i^l < T_i^m; l, m \in A \\ l: \Delta w_l = max_{1 \le k \le n(A)}(\Delta w_k), & T_i^l = T_i^m \end{cases} \quad (4)$$

In the above equation, Δw_l is free space in congestion window w_l, while $n(A)$ is the number of sub-flows in the active set A. To reiterate, the scheduler selects a sub-flow from the active set providing the expected arrival time for the segment on the sub-flow is the shortest. In the event several sub-flows offer equal shortest expected arrival time, the sub-flow with maximum available congestion window space is chosen. When all is equal, the tie is broken randomly.

In addition to scheduling new segments, MPTCP, just like the traditional TCP, performs retransmission of lost segments to improve reliability. Similar to a reordered segment, a lost segment causes a gap in the received segment stream. The gap is filled by retransmitting the lost segment. Meanwhile, ordered higher sequence number segments are kept in the receiver buffer until the lost segment can be successfully retransmitted. When more of the higher sequence number segments arrive and exhaust the receiver buffer space, transmission on the other sub-flows is blocked until there is enough space in the receiver buffer. This leads to reduced throughput and low utilization of the available capacity [101]. Fig. 6.3 illustrates the receiver buffer blocking problem occurring due to a missing lower sequence number segment. In the figure, data segments 2-5 have been received successfully by the receiver. However, they may not be delivered to the higher layers without data segment 1, which is lost on sub-flow 1. The segments are kept in the receiver buffer waiting for segment 1 to be retransmitted. In the case of a full receiver buffer such as in Fig. 6.3, any potential transmissions on sub-flow 2 cannot happen until the lost segment has been retransmitted, hence the receiver buffer blocking problem.

Retransmitting the lost segment on the same sub-flow may not yield desirable recovery from the blocking problem as the sub-flow may still be underperforming. A better solution could be retransmitting on a different sub-flow altogether, provided the new sub-flow can perform better. In this work, it is proposed that the retransmission of the lost segment be on the sub-flow with the current shortest expected arrival time to expedite recovery for applications that cannot wait too long for retransmissions. The sub-flows' transfer times are estimated as in equation (2). It is proposed that the retransmitted segment be treated with higher transmission priority than new segments waiting in the send buffers. The reason is that the retransmitted segment may be blocking the receiver buffer and freeing the buffer for newer segments must be the number one priority. In fact, transmitting newer segments to a blocked buffer would not make any performance sense at all. In the event multiple sub-flows offer the same service times, the lowest loss-rate sub-flow is the preferred choice for the retransmission. The tie is broken randomly.

Sub-flow selection mechanism for retransmitted segments can therefore be expressed as

$$subflow^* = \begin{cases} l, & T_i^l < T_i^m; l, m \in A \\ l: L_l = min_{1 \le k \le n(A)}(L_k), & T_i^l = T_i^m \end{cases} \quad (5)$$

In equation (5), L_l is the estimated loss rate on sub-flow l. The sub-flows' loss rates can easily be estimated by the MPTCP sender keeping a record of segments assumed lost, dividing the result by the total segment load on the sub-flow. For more advanced and accurate loss rate estimation, the technique presented in [loss-rate] can be used. It must be noted that loss rate estimation is not the focus of this research.

An algorithm illustrating complete functionality of the proposed scheduling process, which includes scheduling retransmission of lost segments, is shown in Algorithm IV. The computational complexity of the proposed scheduler is O(n), as *n* sub-flows in the active set have to be examined to select the optimal sub-flow for the incoming segment. In fact, the processing time of the sub-flow selection process for each segment increases with the number of sub-flows. For all practical purposes, the number of sub-flows for a typical modern communication device—such as a smart phone—is not expected to be as large as 10.

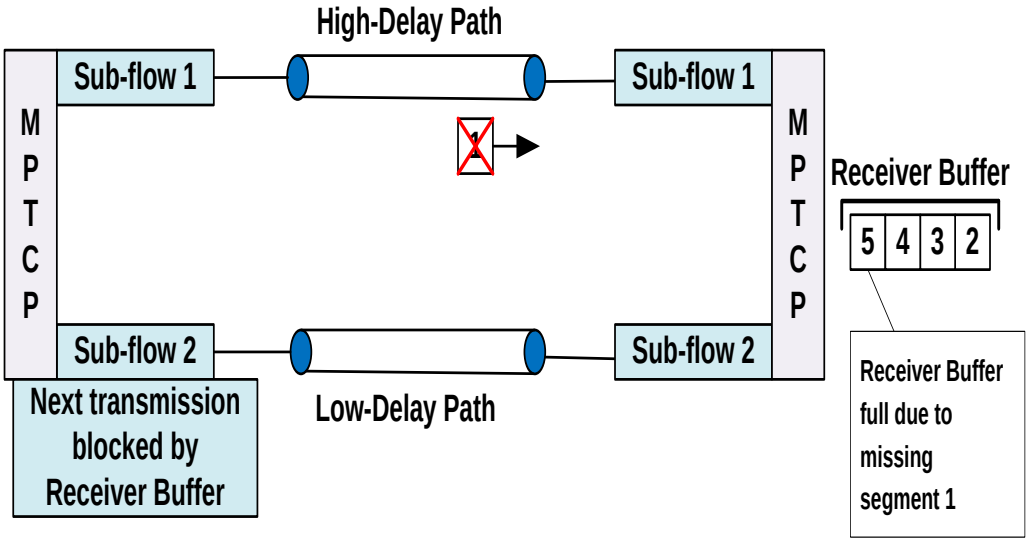

Fig. 6.3 Receiver Buffer Blocking

Algorithm IV Proposed MPTCP Segment Scheduler

1: **Input**:A: a set of sub-flows with free space in the congestion window.

DSN_i: new segment to schedule

2: **if** DSN_i *is not a retransmission* **then**

3: $T_i^l \leftarrow \sum_{k=1}^{i-1} t_k^l + t_i^l$; //$DSN_i's$ estimated transfer time on sub-flow l

4: $$subflow^* = \begin{cases} l, & T_i^l < T_i^m; l, m \in A \\ l: \Delta w_l = max_{1 \le k \le n(A)}(\Delta w_k), & T_i^l = T_i^m \end{cases}$$

5: Enqueue DSN_i into transmission buffer of $subflow^*$;

6: **if** DSN_i *is a retransmission* **then**

7: $$subflow^* \leftarrow \begin{cases} l, & T_i^l < T_i^m; l, m \in A \\ l: L_l = min_{1 \le k \le n(A)}(L_k), & T_i^l = T_i^m \end{cases};$$

8: Enqueue DSN_i as head of the queue in $subflow^*$

9: **end if**

6.3 Re-sequencing at the MPTCP Receiver

The proposed MPTCP segment scheduler may not perfectly avoid segment reordering while striping the segments across multiple sub-flows. To resolve any residual segment reordering at the MPTCP receiver, a re-sequencing buffer is employed. The buffer holds out-of-order segments until the segment that resolves the reordering arrives. When segments are retrieved from multiple sub-flows, the receiving function at the MPTCP level determines whether or not the segments violate the anticipated segment sequence. If the segments are in correct order, they are immediately forwarded to higher layers. Otherwise, the segments are buffered until segment reordering can be resolved. The default MPTCP re-sequencing functionality, however, does not include a mechanism to detect losses and react to them quickly. The MPTCP sender assumes a segment is lost if the retransmission timer expires without receiving an ACK for the segment. The retransmission timer expiration can have devastating effects on MTCP performance since it causes

the congestion window to be set to one segment and then increased conservatively with slow start, leading to drastically reduced application throughput. To circumvent this, a segment loss detection mechanism is added to the MPTCP functionality. The loss detection mechanism must detect losses and act on them quickly enough to avoid the retransmission timer expiration. It may not always be possible for the MPTCP receiver to discern loss in the received segment stream.

The proposed loss detection mechanism is implemented as an extension to the re-sequencing procedure at the MPTCP receiver. To determine segment reordering at the MPTCP receiver, a variable ($\boldsymbol{DSN_e}$) is maintained to track the data sequence number of the anticipated segment. Another variable, $\boldsymbol{DSN_i}$, is used to point to the arriving segment's data sequence number. If $\boldsymbol{DSN_i}$ is equal to $\boldsymbol{DSN_e}$, then the segment has arrived in correct order. This is illustrated in Fig. 6.4. In the figure, three outgoing segments 1-3 are sent over sub-flow 1 and sub-flow 2. Segments 1-2 have been received in correct order, and they can be delivered to higher layers immediately. Segment 3, which is about to be retrieved from sub-flow 2, is also arriving in the expected order, which is desirable for optimal MPTCP performance. In the event the arriving segment is one that is expected to resolve segment reordering, the segment is delivered to the application together with $\boldsymbol{B_{DSNe}}$—a block of ordered higher sequence number segments that have been waiting for $\boldsymbol{DSN_e}$ to arrive.

If $\boldsymbol{DSN_i}$ is greater than $\boldsymbol{DSN_e}$, then the arriving segment is out of order and must be kept in the buffer until the reordering can be resolved. For example, in Fig. 6.5, the expected sequence number is $\boldsymbol{DSN_e} = 1$. However, the segment that has just arrived is $\boldsymbol{DSN_i} = 2$, which leads to a reordering event. If $\boldsymbol{DSN_i}$ from all the available interfaces does not match $\boldsymbol{DSN_e}$, then the expected segment is assumed lost. This assumption makes sense since higher sequence number segments are not expected to overtake lower sequence number segments along the same sub-flow. Fig. 6.6 is an illustration of this. When loss has been detected, the MPTCP receiver must send a DUPACK immediately to trigger retransmission of the lost segment without waiting for a timeout, which can degrade performance adversely. Algorithm V illustrates the operation of the re-sequencing by the MPTCP receiver as proposed in the thesis. The upcoming section discusses the performance of the proposed MPTCP scheduler. The performance of the scheduler is compared to the LowRTT and the round robin mechanisms.

Algorithm V Re-sequencing Unit

1: Arriving segment (DSN_i) is read from sub-flow

2: **if** ($DSN_i == DSN_e$) **then**

3: Forward DSN_i & B_{DSN_e} to the application;

4: **else if** $DSN_i > DSN_e$ **then**

5: Insert DSN_i in the Reorder Buffer;

6: **end if**

7: **if** $DSN_{i,\forall l} > DSN_e$ **then**

8: DSN_e is lost

9: Send DUPACKs for the loss

10: **end if**

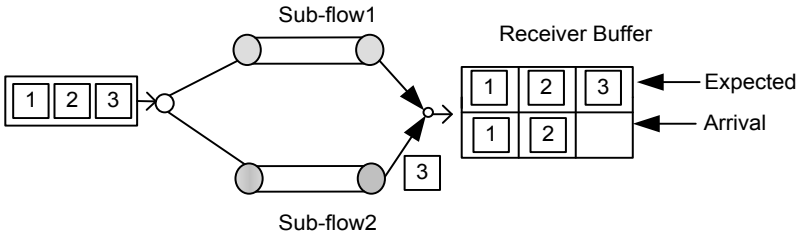

Fig. 6.4 In-order Segment Delivery

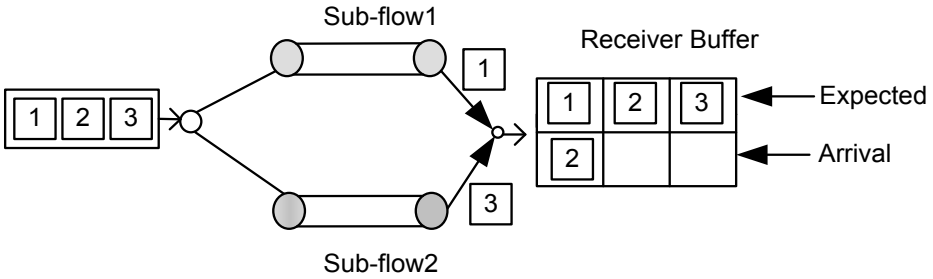

Fig. 6.5 Out-of-order Segment Delivery

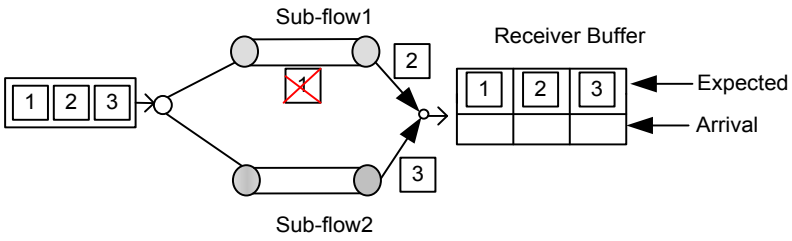

Fig. 6.6 Segment Loss

6.4 Experimental Setup and Analysis

Performance of concurrent multipath transport protocols, such as MPTCP, is known to be affected by network path heterogeneity. The difference in delay, bandwidth and loss between the different network paths can significantly decrease MPTCP throughput performance. It is, therefore, imperative to equip MPTCP with suitable functionality to adapt to network path heterogeneity so that the benefits of MPTCP can fully be realized. The focus of this research is on scheduling to improve throughput performance of MPTCP. A scheduler, which has been described above, has been proposed. In this section, the impact of the scheduler on MPTCP performance is evaluated against two reference schedulers: LowRTT and Round Robin as introduced earlier.

6.4.1 Network Topology

The network topology considered in the simulation experiments is a client-server model, where the client and the server are connected through three heterogeneous network paths. In the first set of the simulation experiments, only two network paths are activated between the client the server. In the second set, all the three network paths are enabled. MPTCP traffic is generated by transmitting a 10MB data file from the server to the client. The network paths between the client and the server are configured with some degree of heterogeneity in terms of delay and loss. Moreover, a scenario with varying receiver buffer levels is evaluated. Fig. 6.7 depicts the simulation topology. It should be noted that, even though the simulation experiments that have been carried out consider up to three network paths, the results of the proposed scheduling mechanism can be generalized to a larger number of network paths.

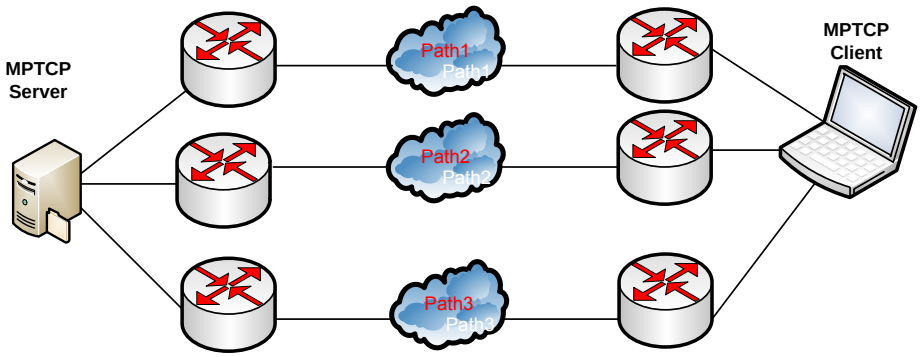

Fig. 6.7 Simulated Network Topology

6.4.2 Simulation Environment

To evaluate the MPTCP schedulers, the ns-3 simulator has been used. The ns-3 simulator natively supports multi-homing, which greatly simplifies the simulation of concurrent multipath transmission. The ns-3, however, does not include the MPTCP functionality. The MPTCP module developed in [102] was ported into the ns-3 simulator. The MPTCP patch already included an implementation of the round robin MPTCP scheduler within the *GetSubflowToUse()* function, so only the proposed MTPCP scheduler and the LowRTT had be implemented from scratch in C++ and then appropriately incorporated into the MPTCP *GetSubflowToUse()* routine.

A 10MB file was transferred from the server to the client under different heterogeneity configurations in terms of delay and loss. The performance of the schedulers was also evaluated with different receiver buffer settings. In the first set of simulation experiments, two network paths were considered between the server and the client. In the second set, three network paths were considered to evaluate the impact of increasing the number of paths on MPTCP performance.

The results of the simulation experiments are discussed in the next subsection. The focus of the experiments is on the throughput goal of MPTCP. Each set of results is an average of five simulation runs, each running for 2 minutes.

6.4.3 Experimental Scenarios and Results

A. Two-path multipath network with varying receiver buffer size

The receiver buffer can have profound impact on the performance of MTCP. In the event of high segment reordering, a constrained receiver buffer may not be able to accommodate all the reordered segments, which may lead to buffer overflow induced segment loss. An ideal buffer size for MPTCP to realize satisfactory aggregation benefit is the sum of the paths' bandwidth-delay product [103], which may not always be practical. A desirable MPTCP scheduler should be able to perform well even if the receiver buffer is constrained.

In this experiment, the network paths' bandwidth is fixed at 1 Mpbs. The propagation delay of Path1 is fixed at 50 ms while the delay on Path2 is 100 ms, thus creating some amount of delay heterogeneity between the network paths. The receiver buffer is varied from 25 KB to 125 KB. Fig. 6.8 shows the performance of the investigated MPTCP schedulers in terms of throughput. It can be observed that the throughput performance of the proposed MPTCP scheduler remains high even at very low receiver window levels. On the other hand, the reference schedulers experience lower throughput, which only increases with the buffer size. Even though the LowRTT performs poorer than the proposed MPTCP scheduler does, it outperforms the round robin. This can be attributed to the round robin's worst handling of the delay difference between the network paths. Without proper handling of the delay difference, high segment reordering ensues, causing a significant drop in throughput under constrained buffer conditions.

The proposed MPTCP scheduler's superior performance can be attributed to the manner in which segments are striped across the different network paths. To recapitulate, the proposed MPTCP scheduler assigns a segment to the sub-flow that promises the shortest expected arrival time. This makes it less likely for succeeding segments on any sub-flow to overtake the current segment, which effectively avoids segment reordering or leads to the least segment reordering. Consequently, the proposed MPTCP scheduler requires the smallest receiver buffer of the three evaluated schedulers. The round robin, which is more prone to high segment reordering, requires the largest receiver buffer space to resolve the packet reordering. MPTCP schedulers, which are prone to segment reordering tend to be less efficient in utilizing the aggregated bandwidth capacity. Fig. 6.9 depicts the efficiency of the evaluated schedulers. As expected, the proposed MPTCP scheduler is the most efficient, whereas the round robin is the least efficient.

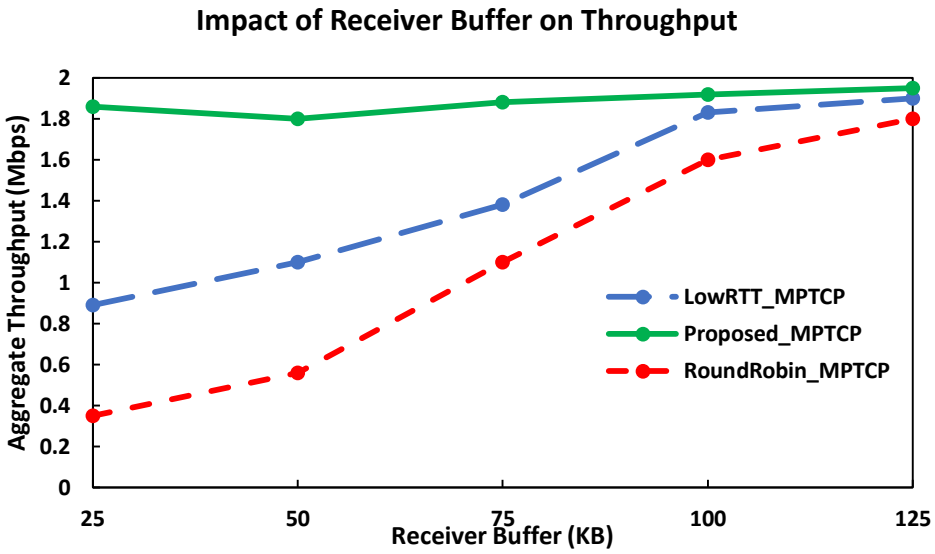

Fig. 6.8 Impact of Receiver Buffer on MPTCP Throughput

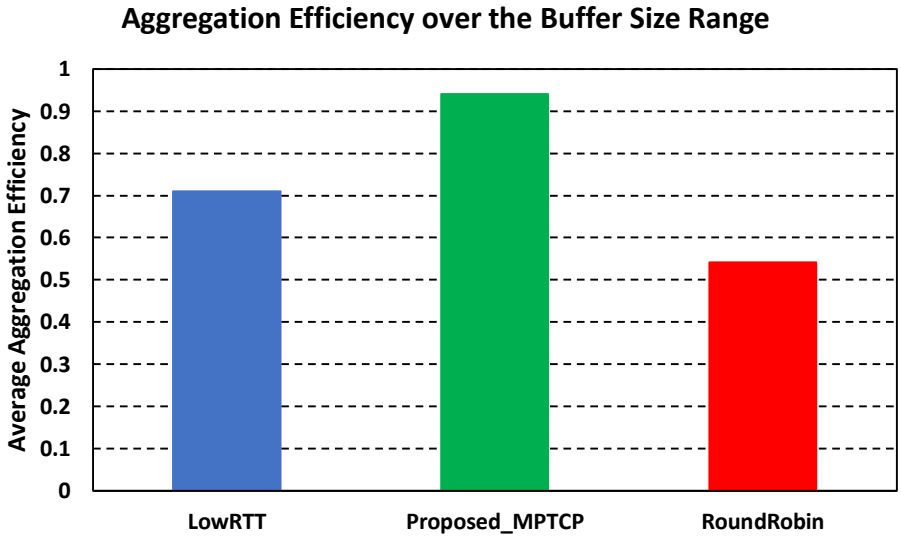

Fig. 6.9 Aggregation Efficiency over the Buffer Size Range

B. Delay Heterogeneity

In this part of the experiment, the different schedulers have been investigated under delay heterogeneity. The delay heterogeneity was created by varying the delay on one path and fixing it on the rest of the network paths. In the first set of experiments, where two network paths were considered, the delay on Path1 was set at 50ms while Path2's delay varies from 50ms to 250ms. In the second set of experiments, the delay on Path1 and Path2 was set at 50ms and 100ms,

respectively. The delay on Path3 was varied from 50ms to 250ms. The bandwidth for all the paths was set at 1 Mbps in all experiments. The aim of this experiment was to investigate the impact of increasing delay heterogeneity on MPTCP performance and to determine how the different schedulers react to the heterogeneity in order to maintain good MPTCP performance.

Fig. 6.10 presents throughput results for the two-path MPTCP scenario. It can be observed that the proposed MPTCP scheduler achieves and sustains the highest throughput performance of all the evaluated schedulers. The proposed scheduler does not suffer as high segment reordering as the LowRTT and the round robin scheduling mechanisms. Increasing segment reordering (resulting from increasing delay heterogeneity) in the case of LowRTT and the round robin causes increased spurious retransmissions, which hurt MPTCP performance. Segment reordering increases with the number of heterogeneous paths, which results in increased spurious retransmission for MPTCP. Therefore, MPTCP performance for schedulers that do not resolve segment reordering well worsens as the number of heterogeneous paths increases. This is confirmed by the throughput results shown in Fig. 6.11. In the figure, MPTCP performance is resilient to delay heterogeneity in the three-path scenario under the proposed scheduler. Performance under LowRTT and round robin has worsened with the increase in number of paths.

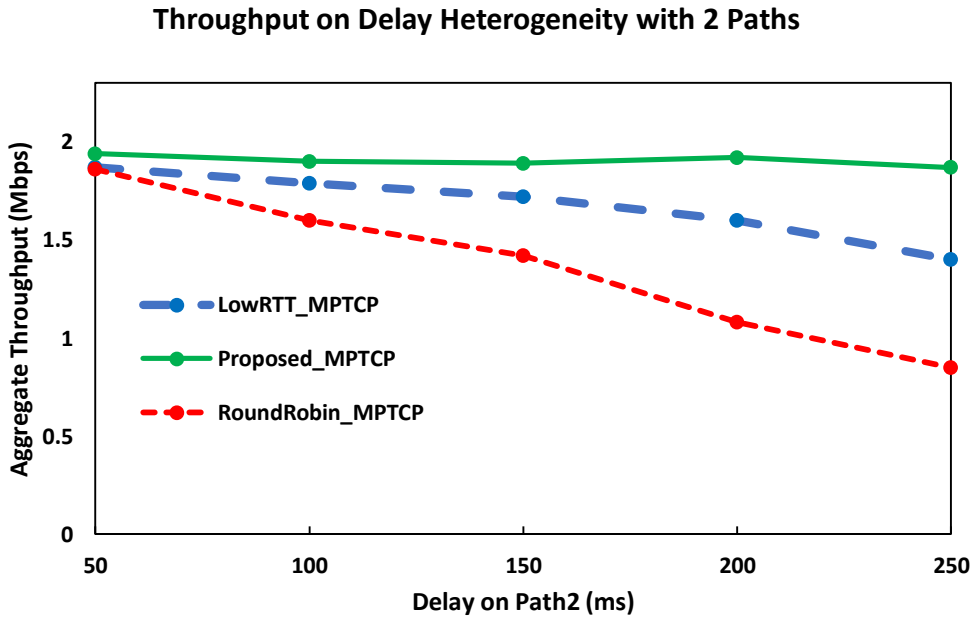

Fig. 6.10 Delay Heterogeneity on two paths

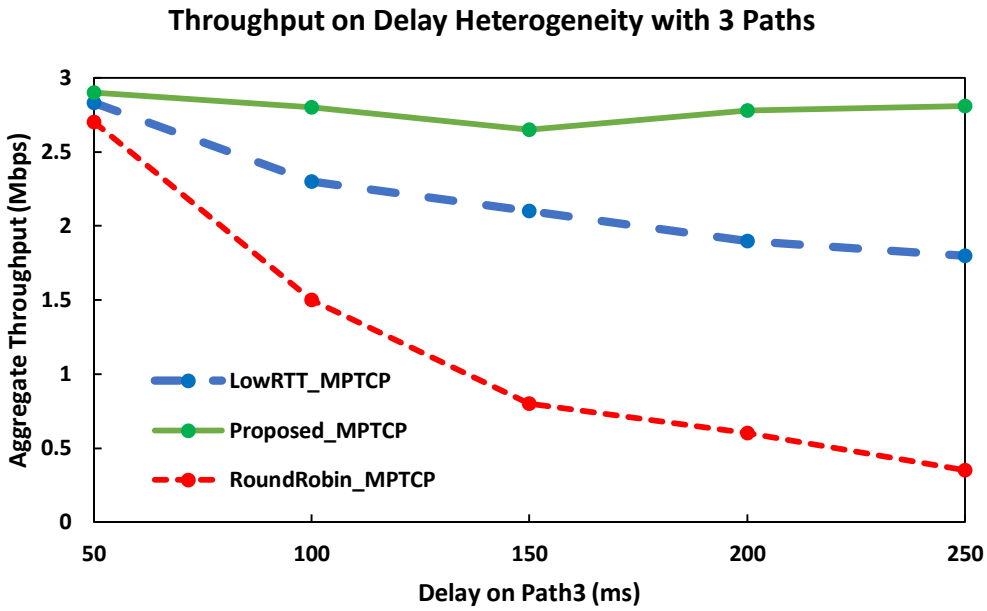

Fig. 6.11 Delay Heterogeneity on three paths

C. Loss Heterogeneity

In the experiment, a two-path MPTCP environment was considered. The two paths' bandwidth was set at 1 Mbps each. The loss rate on Path1 was set at 1%, while on Path2 the loss rate was varied from 0% to 5%. The receiver buffer seize was set to equal the paths' bandwidth-delay product. The delay on Path1 was set at 50ms while the delay on Path2 was 100ms. The purpose of the experiment was to evaluate MPTCP's robustness to loss and small fixed delay disparity under the scheduling frameworks studied in the thesis. The throughput results for the three scheduling mechanisms under loss heterogeneity are depicted in fig 6.12. Generally, MPTCP throughput decreases with the increase in loss rate on path2 for all the investigated schedulers. However, the proposed scheduler's throughput decreases more gradually than the LowRTT and the Round Robin. Whenever possible, the proposed scheduling framework is able to detect losses early enough to avoid the effect of retransmission timer expiration, which the reference scheduling schemes are not able to avoid. Reacting to losses by waiting for retransmission timeout leads to throttling of the sender's sending rate, thus leading to reduced throughput, as it is the case with the LowRTT and Round Robin mechanisms.

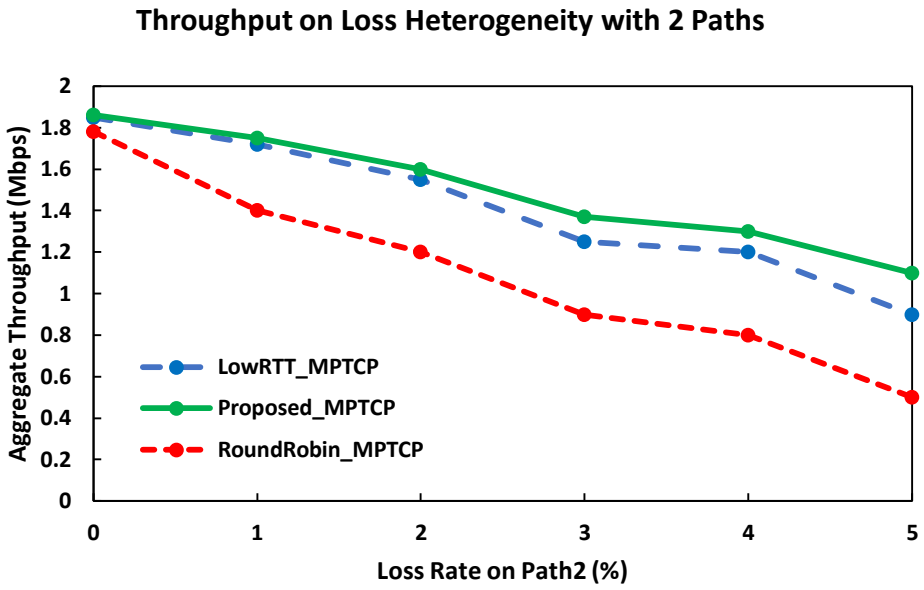

Fig. 6.12 Loss Heterogeneity on two paths

6.5 Chapter Summary

The chapter presented concurrent multipath transmission based on the IETF Multipath TCP and investigated the effect of using the default schedulers (the LowRTT and the Round Robin) and the new scheduling mechanism, which has been proposed in this thesis. The key feature of the proposed scheduler is adaptation to path heterogeneity to maximize aggregation benefit in terms of improved throughput performance. The investigated scheduling mechanisms were evaluated using a series of ns-3 simulation experiments. The results showed that the proposed scheduling solution is more robust to path heterogeneity than the reference solutions as it achieved and sustained high throughput performance under various network path heterogeneity configurations.

The studied scheduling mechanisms were also evaluated under constrained receiver buffer conditions. For different receiver buffer levels, the throughput performance under the proposed scheduler was barely degraded. Meanwhile, the reference scheduling solutions only achieved improved throughput performance as the size of the receiver buffer was increased.

Chapter 7

Conclusion and Future Work

This chapter provides a summary of the research contributions presented in this thesis and discusses interesting open research areas for future work.

7.1 Summary of Contributions

Emerging user's devices come equipped with multiple network interfaces, enabling the devices to connect to several networks at the same time, which makes ubiquitous access a reality. Furthermore, new possibilities—such as aggregating bandwidth from the different network interfaces to improve performance—are possible. However, the problem of bandwidth aggregation is challenging due to the heterogeneous nature of the different networks that a device with multiple network interfaces can connect to. Ignoring network heterogeneity in bandwidth aggregation can lead to undesirable effects, such as packet reordering and poor load balancing, which can degrade performance of applications that use concurrent multipath transmission. The primary goal of this thesis was to study the problem of bandwidth aggregation and propose efficient techniques to address the challenges of bandwidth aggregation. The research contributions of the thesis are summarized as follows:

1. Critical and comprehensive review of existing bandwidth aggregation approaches

   The first important research contribution presented in the thesis is a comprehensive and critical review of bandwidth aggregation solutions that have been proposed in the literature to enable concurrent multipath transmission for multi-homed user's devices in heterogeneous networks. The review provides useful information for researchers and developers in the area of bandwidth aggregation. The different bandwidth aggregation mechanisms have been classified according to two criteria: adaption to network conditions and layer of the network protocol stack at which a bandwidth aggregation solution is realized.

With regard to adaption to network conditions, a bandwidth aggregation solution can either be adaptive or non-adaptive. An adaptive bandwidth aggregation solution takes note of varying network characteristics, such as bandwidth, delay and loss rate, and appropriately adapts its concurrent multipath transmission strategy to improve performance. The network protocol stack layers that make the layer-based classification are application, transport, network, and data link layers. The proposed classification of the different bandwidth aggregation techniques provides the basis to discern the operation, strengths and shortcomings of each of the reviewed techniques. The observation of the shortcomings of existing bandwidth aggregation approaches motivated the research carried out in this thesis.

2. Delay-based multipath packet scheduler to combat packet reordering

One of the challenges of bandwidth aggregation for concurrent multipath transmission is packet reordering, which occurs when packets of the same application get to the receiver out of the correct order. That is, the higher sequence number packet arrives before the lower sequence number packet. Packet reordering hurts application performance by causing additional delay, which may not be tolerated by real-time applications. Packet reordering can also affect TCP-based applications adversely by triggering unnecessary retransmissions and throttling of the congestion window.

This thesis presented an efficient network layer multipath packet scheduler that stripes packets of the same application across multiple network paths in a way that effectively minimizes packet reordering. The scheduler considers end-to-end delay on the different network paths and determines the delay difference between the network paths. Considering delay difference helps to allocate appropriate traffic load across each of the network paths for proper load balancing. It also helps to forecast the packets' transmission times on the different network paths so that the packets can be scheduled on the network paths to arrive at the receiver in correct order. The proposed multipath packet scheduler has been evaluated in the ns-3 simulation environment under bandwidth and delay disparity across the different network paths. The metrics used to evaluate the scheduler included reordering delay, reorder buffer overflow and reorder density. The results have shown that the proposed multipath packet scheduler outperforms the considered reference schedulers.

3. Multipath streaming of H.264 SVC video-on-demand

Video streaming applications have become dominant in the Internet, making up the largest portion of the Internet traffic. Video is consumed by users with a wide range of devices, which include smart phones that come equipped with multiple network interfaces. Sometimes a single network interface on a smartphone may not be enough to enable the streaming of high definition video, which requires large amount of bandwidth to meet the target bitrate. One way to increase the required bandwidth is to aggregate bandwidth from the different network interfaces for concurrent multipath transmission of demanding video applications.

In this thesis, a bandwidth aggregation solution to stream H.264 SVC video-on-demand over multiple network paths has been proposed. The proposed multipath video streaming solution adaptively assigns the SVC video layers to a set of network paths with the lowest loss rate and enough aggregated bandwidth to meet the required video bitrate. When multiple network paths have been assigned to a SVC video layer, the layer's video packets are intelligently striped across the network paths for concurrent transmission, ensuring that the video packets can meet their playback deadlines. Experimental results have shown that the proposed concurrent multipath video streaming solution outperforms reference streaming solutions from previous work in terms of received video quality measured in terms of PSNR and SSIM.

4. Scheduling Framework to Improve Performance of the IETF Multipath TCP

Concurrent multipath transmission can be accomplished at different layers of the network protocol stack. The IETF has proposed extensions to TCP to leverage multiple network paths between the sender and the receiver in order to improve performance at the lowest cost possible. Segment scheduling, which allows for data segments to be striped across multiple sub-flows is one of the important functions making up the IETF Multipath TCP. A multipath TCP scheduler must be able to adapt to the conditions of the different network paths in order to realize the benefit of aggregating bandwidth from multiple network paths. Such a scheduling mechanism has been proposed in this thesis. The rational of the proposed scheduling method is to schedule a data segment on a network path that promises the shortest expected arrival time. This effectively lowers segment reordering and improves MPTCP

throughput performance. The proposed solution also includes a mechanism to detect segment losses early enough to avoid retransmission timeouts, which can have drastic impact on throughput performance of MPTCP. An evaluation framework in the ns-3 simulation environment has been developed to evaluate the efficacy of the proposed solution. The results have shown that the solution can achieve higher throughput performance than the default MPTCP schedulers under various path heterogeneity settings and different receiver buffer levels.

7.2 Future Work

In this thesis, efficient bandwidth aggregation solutions have been proposed to enable multi-homed devices to improve application performance through concurrent multipath transmission in heterogeneous networks. However, there are still interesting areas of future work that need to be explored. The areas of future work are discussed as follows:

1. **Optimization of Battery Power Consumption**: One of the challenges of concurrent multipath transmission affecting handheld devices has always been limited battery power. The device's battery power consumption when a single network interface is active is significant and it can get depleted quite quickly. The use of multiple network interfaces makes the problem of battery power consumption even worse, which puts the handheld device at much higher risk of premature termination of ongoing communication sessions. Research work such as in [18] must be done to develop efficient bandwidth aggregation solutions that include mechanisms to prolong the device' battery power so that the benefits of concurrent multipath transmission can be fully realized.

2. **Test-bed Implementation and Evaluation of Concurrent Multipath Transmission**: Many bandwidth aggregation techniques—including the solutions proposed in this thesis—have mostly been evaluated in the simulation environment. Even though the simulation results can sufficiently validate the efficacy of the proposed bandwidth aggregation solutions, test-bed implementation and evaluation are necessary to ascertain the practicality of these solutions.

3. **Concurrent Multipath Transmission for Multiple Applications**: The bandwidth aggregation solutions developed in this thesis were based on a single application striping packets across multiple network paths for concurrent multipath transmission. However, a more realistic scenario is one where a multi-homed device runs several applications that must share the aggregated bandwidth fairly. More research needs to be done to improve existing solutions to enable concurrent multipath transmission for multiple applications running on a device that is equipped with multiple network interfaces.

4. **Effect of Mobility on Concurrent Multipath Transmission**: In a heterogeneous wireless network, a multi-homed device can move across the network, losing connectivity to some of the networks and discovering new ones. In the thesis, the proposed bandwidth aggregation solutions were evaluated in static scenarios. More realistic scenarios that better capture the effect of mobile device mobility on bandwidth aggregation need to be explored.

References

[1] Cisco Visual Networking Index: Forecast and Methodology. Accessed from: www.cisco.com/c/en/us/solutions/collateral/service-provider/ip-ngn-ip-next-generation-network/white_paper_c11-481360.pdf

[2] Zinner T, Tutschku K, Nakao A and Tran-Gia P, "Using concurrent multipath transmission for Transport Virtualization: Analyzing path selection," 22nd International Teletraffic Congress (ITC), 7-9 Sept. 2010

[3] Golubchik L, Lui JCS, Tung TF, ALH Chow , Lee A, Franceschinis G and Anglano C, "Multi-path continuousmedia streaming: What are the benefits?", Perform.Eval., vol. 49, no. 1/4, pp.429 -449 2002

[4] Evensen K, Kaspar D and Griwodz C, "Using Bandwidth Aggregation to Improve the Performance of Quality-adaptive Streaming," Elsevier Signal Processing: Image Communication (2011)

[5] Frossard P, De Martin JC and Civanlar MR, "Media Streaming With Network Diversity," in Proceedings of the IEEE , vol.96, no.1, pp.39-53, Jan. 2008

[6] Qadir J, Ali A, Kok-Lim Y, Sathiaseelan A and Crowcroft J, "Exploiting the power of multiplicity: a holistic survey of network-layer multipath,"in Communications Surveys & Tutorials, IEEE , vol.PP, no.99, pp.1-1

[7] Pawar P, van Beijnum B, Peddemors A and van Halteren A, "Context-Aware Middleware Support for the Nomadic Mobile Services on Multi-homed Handheld Mobile Devices," 12th IEEE Symposium on Computers and Communications, July 2007, pp. 341-348

[8] Arthur CM, Lehane A and Harle D, "Keeping Order: Determining the Effect of TCP Packet Reordering," Third International Conference on Networking and Services, vol., no., pp.116-116, 19-25 June 2007

[9] Gao C, Ling Z and Yuan Y, "Packet Reordering Analysis for Concurrent Multipath Transfer, " International Journal of Communication Systems, 2013, vol. 27, pp. 4510-4526.

[10] Abley J, Lindqvist K, Davies E, Black B and Gill V, "IPv4 Multihoming Practices and Limitations, " Internet RFC 4116 (informational), July 2005

[11] Khattab O and Alani O, "Overview of Interworking Architectures in Heterogeneous Wireless Networks: Objectives, Features and Challenges",

[12] Bazzi A, Pasolini G, Andrisano O, "Multiradio Resource Management: Parallel Transmission for Higher Throughput?", EURASIP Journal on Advances in Signal Processing 2008.

[13] Wu J, Cheng B, Yuen C and Shang Y, "Distortion-aware concurrent multipath transfer for mobile video streaming in heterogeneous wireless networks," Mobile Computing, IEEE Transactions on 14.4 (2015): 688-701.

[14] Zheng K, Jiao X, Liu M and Li Z, "An analysis of resequencing delay of reliable transmission protocols over multipath", IEEE International Conference on Communications (ICC), 2010.

[15] Miu KA. Improving Packet Delivery Efficiency Using Multi-Radio Diversity in Wireless LANs. MIT. Thesis, 2006

[16] Hong Cheng, Yaohui Jin, Yu Gao, Ying Di Yu, Weisheng Hu and Ansari N, "Per-Flow Re-Sequencing in Load-Balanced Switches by Using Dynamic Mailbox Sharing, " IEEE International Conference on Communications (ICC), 2008; 5680-5684

[17] Venkatasubramanian S and Gopalan NP, "QOS based Robust multipath routing protocol for mobile adhoc networks", IACSIT International Journal of Engineering and Technology, 2009, 1(5).

[18] Mahkoum H, Sarikaya B and Hafid A, "A Framework for Power Management of Handheld Devices with Multiple Radios", In Proceedings of IEEE WCNC, 2009: 1-6.

[19] Przybylski M, Belter B and Binczewski A, "Shall we worry about packet reordering", Computational Methods in Science and Technology, 2005, 11(2); 141–146

[20] Bennett J.C.R, Partridge C and Shectman N, "Packet reordering is not pathological network behaviour", IEEE/ACM Transactions on Networking, 1999, 7(6); 789-798.

[21] Chowdhury M.Z, Yeong Min Jang, Choong Sub Ji, Sunwoong Choi, Hongseok Jeon, Junghoon Jee and Changmin Park, "Interface selection for power management in UMTS/WLAN overlaying network", In Proceedings of ICACT, 2009, 1; 795-799.

[22] Jayasumana A, Piratla N,Banka T,Bare A and Whitner R, "Improved Packet Reordering Metrics", RFC 5236, 2008.

[23] Piratla, Nischal M and Anura P Jayasumana. "Metrics for packet reordering—A comparative analysis." International Journal of Communication Systems 21.1 (2008): 99-113.

[24] Chebrolu K and Rao R, "Communication using multiple wireless interfaces", In Proceedings of IEEE WCNC, 2002; vol.1, pp. 327- 331.

[25] Wang X, Feng Z, Fan D, Xue Y and Le V, "A Segment-Based Adaptive Joint Session Scheduling Mechanism in Heterogeneous Wireless Networks", In Proceedings of IEEE VTC, 2009: 1-5.

[26] Hari A, Varghese G and Parulkar G, "An Architecture for Packet-striping Protocols", ACM Transations on Computer Systems, 1999; 17(4).

[27] Sharma P, Lee S.-J, Brassil J and Shin K. G, "Distributed channel monitoring for wireless bandwidth aggregation", In Proceeding of the IFIP-TC6 Networking, 2004: 345–356.

[28] Allman M, Kruse H and Ostermann S, "An Application-Level Solution to TCP's Satellite Inefficiencies, " In Workshop Proceedings on Satellite-based Information Services, 1996

[29] Allcock W, "Gridftp: Protocol extensions to ftp for the grid", In Global Grid ForumGFD-R-P.020, 2003.

[30] Stewart R, Ramalho M, Xie Q, Tuexen M and Conrad P, "Stream Control Transmission Protocol Partial Reliability Extension", RFC 3758, May 2004.

[31] Ford A, Raiciu C, Handley M and Bonaventure O, "Extensions for Multipath Operation with Multiple Addresses", RFC 6824, Jan 2013.

[32] Sklower K, Lloyd G, McGregor D and Coadetti T, "The PPP Multilink Protocol", RFC 1990, 1996.

[33] Sachs J, "A generic link layer for future generation wireless networking", In Proceedings of IEEE ICC, 2003, 2; 834- 838.

[34] Sivakumar H, Bailey S and Grossman S, "PSockets: The case for application-level network striping for data intensive applications using high speed wide area networks", In Proceedings of ACM/IEEE SC, 2000.

[35] Mohamed N, Al-Jaroodi J, Jiang H and Swanson D, "A user-level socket layer over multiple physical network interfaces", In Proceedings of the International Conference on Parallel and Distributed Computing and Systems, 2002:810–815.

[36] Ford A, Raiciu C, Handley M, Barr´e S and Iyengar J, "Architectural Guidelines for Multipath TCP Development", IETF RFC 6182, March 2011.

[37] Barre S, Bonaventure O, Raiciu C and Handley M, "Experimenting with Multipath TCP", In Proceedings of ACM SIGCOMM, 2010.

[38] Nguyen SC, Zhang X, Nguyen Thi, Mai Trang and Pujolle G, "Evaluation of throughput optimization and load sharing of multipath TCP in heterogeneous networks", In Proceedings of Wireless and Optical Communications Networks, 2011:1-5.

[39] Zhang Y, Wang C and Gao Y, "Weighted Size-Aware Packet Distribution for Multipath Live Streaming", In Proceedings of IEEE ICC, 2009: 1-5.

[40] lKaspar D, Evensen K, Hansen AF, Engelstad P, Halvorsen P and Griwodz C, "An Analysis of the Heterogeneity and IP Packet Reordering over Multiple Wireless Networks", In IEEE Symposium on Computer and Communications, 2009; 637-642.

[41] Kim PS, Yoon Y and Kim H, "A packet partition scheduling mechanism for bandwidth aggregation over multiple paths", Journal of Convergence Information Technology, 2008; 3(4):325-41.

[42] Koudouridis GP, Agüero R, Alexandri E, Choque J, Dimou K, Karimi HR, Lederer H, Sachs J and Sigle R, "Generic Link Layer Functionality for Multiradio Access Networks", In Proceedings of Mobile Summit , 2005.

[43] Yaver A and Koudouridis G.P, "Performance Evaluation of Multi-Radio Transmission Diversity: QoS Support for Delay Sensitive Services", In Proceedings of IEEE VTC, 2009: 1-5.

[44] Luo J, Mukerjee R, Dillinger M, Mohyeldin E and Schulz E, "Investigation of radio resource scheduling in WLANs coupled with 3G cellular network", IEEE Communications Magazine, 2003, 41(6); 108- 115.

[45] Sharma P, Lee S, Brassil J and Shin KG, "Aggregating Bandwidth for Multihomed Mobile Collaborative Communities", IEEE Transactions on Mobile Computing, 2007, 6(3); 280-296.

[46] Orozco-Barbosa L and Taeseop H, "On the use of frame-based slice size for the robust transmission of MPEG video over ATM networks", IEEE Transactions on Broadcasting, 2000, 46(2); 134-143.

[47] Ekmekci S and Sikora T, "Multistate vs. single-state video coding over error-prone channels", Record of the Thirty-Seventh Asilomar: Signals, Systems and Computers, 2003, 2; 1544- 1547.

[48] Xue Y, Lin Y, Feng Z, Cai H and Chi C, "Autonomic Joint Session Scheduling Strategies for Heterogeneous Wireless Networks", In Proceedings of IEEE WCNC, 2008; 2045-2050.

[49] Qureshi A and Guttag J, "Horde: Separating Network Striping Policy from Mechanism", In Proceedings of ACM MobiSys, 2005.

[50] Kaspar D, Evensen K, Engelstad P, Hansen AF, Halvorsen P and Griwodz C, "Enhancing Video-on-Demand Playout over Multiple Heterogeneous Access Networks", In Proceedings of IEEE CCNC, 2010.

[51] Kaspar D, Evensen K, Engelstad P and Hansen AF, "Using HTTP Pipelining to Improve Progressive Download over Multiple Heterogeneous Interfaces", In Proceedings of IEEE ICC, 2010;1-5.

[52] Casetti C and Gaiotto W, "Westwood sctp: load balancing over multipaths using bandwidth-aware source scheduling", In Proceedings of IEEE VTC, 2004.

[53] Fiore M and Casetti C, "An adaptive transport protocol for balanced multihoming of real-time traffic", In Proceedings of IEEE GLOBECOM, 2005.

[54] El A A, Saadawi T and Lee M, "LS-SCTP: a Bandwidth Aggregation Technique for Stream Control Transmission Protocol", Computer Communications, 2004, 27; 1012–1024.

[55] Liao J, Wang J and Zhu X, "cmpSCTP: An Extension of SCTP to Support Concurrent Multi-Path Transfer", In Proceedings of IEEE ICC, 2008.

[56] Hasegawa Y, Yamaguchi I, Hama T, Shimonishi H and Murase T, "Improved data distribution for multipath TCP communication", In Proceedings of IEEE GLOBECOM, 2005.

[57] Hsieh H-Y and Sivakumar R, "A Transport Layer Approach for Achieving Aggregate Bandwidths on Multihomed Mobile Hosts", In Proceedings of ACM MOBICOM, 2002.

[58] Anand J and Sarkar D, "cmpRTCP: Concurrent Multi-Path Real-Time TCP", In Proceedings of IEEE GLOBECOM, 2007.

[59] Mirani FH, Boukhatem N and Minh AT, "A Data-Scheduling Mechanism for Multi-Homed Mobile Terminals with Disparate Link Latencies", In Proceedings of IEEE VTC, 2010.

[60] Mirani F.H, Kherraz M and Boukhatem N, "Forward prediction scheduling: Implementation and performance evaluation", In Proceedings of ICT, 2011.

[61] Mirani FH, Xiaofei Z, Boukhatem N and Thi-Mai-Trang N, "Cross-layer FPS: A SCTP-based cross-layer data scheduling approach", In Proceeding of IEEE CCNC, 2011.

[62] Chebrolu K and Rao R. "Bandwidth aggregation for real-time applications in heterogeneous networks", IEEE Transactions on Mobile Computing, vol. 5, No. 4, April 2006.

[63] Chebrolu K and Rao R, "Selective frame discard for interactive video", In Proceedings of IEEE ICC, 2004.

[64] LIU G, ZHOU X and ZHU G, "A scheduling algorithm for maximum throughput based on the link condition in heterogeneous network", Journal Communication and Computer, 2007.

[65] Taleb T, Fernandez JC, Hashimoto K, Nemoto Y and Kato N, "A Bandwidth Aggregation-Aware QoS Negotiation Mechanism for Next-Generation Wireless Networks", In Proceedings of IEEE GLOBECOM, 2007.

[66] Fernandez JC, Taleb T, Guizani M and Kato N, "Bandwidth Aggregation-Aware Dynamic QoS Negotiation for Real-Time Video Streaming in Next-Generation Wireless Networks", IEEE Transactions on Multimedia, 2009, 11(6); 1082 – 1093.

[67] Chebrolu K, Raman B and Rao R, "A Network Layer Approach to Enable TCP over Multiple Interfaces", Journal of Wireless Networks (WINET), 2005.

[68] Partridge C, "ACK spacing for high delay-bandwidth paths with insufficient buffering", IETF Internet Draft, Sept. 1998, draft-rfced-info-partridge-01.txt.

[69] Prabhavat S, Nishiyama H, Ansari N and Kato N, "Effective Delay-Controlled Load Distribution over Multipath Networks", IEEE Transactions on Parallel and Distributed Systems, 2011, 22(10); 1730-1741.

[70] Lin YH and Tsao S, "Dynamic Bandwidth Aggregation for a Mobile Device with Multiple Interfaces", In Proceedings of ISCOM, 2005.

[71] Ahmad SZ, Qadir MA and Akbar MS, "QoS Optimization of In-elastic Flows Stripped over Multiple Asymmetric Channels in Mobile Networks", In ICUMT, 2009.

[72] Tsai M, Naveen K. Zeadally CS and Shieh C, "A Concurrent Multi-path Transmission Control Scheme to Reduce Packet Reordering Latency at the Receiver", In Advanced Technologies for Communications, 2008.

[73] Evensen K, Kaspar D, Engelstad P, Hansen AF, Griwodz C and Halvorsen P, "A Network-Layer Proxy for Bandwidth Aggregation and Reduction of IP Packet Reordering", In Proceedings of IEEE LCN, 2009.

[74] Evensen K, Kaspar D, Engelstad P, Hansen AF, Griwodz C and Halvorsen P, "Using Multiple Links to Increase the Performance of Bandwidth-Intensive UDP-Based Applications", In Proceedings of IEEE ISCC, 2011.

[75] Zhang Z, Wen X and Zheng W, "A NAT Traversal Mechanism for Peer-To-Peer Networks", In Intelligent Ubiquitous Computing and Education, 2009.

[76] Azad MA, Mahmood R and Mehmood T, "A Comparative Analysis of DCCP Variants (CCID2, CCID3), TCP AND UDP for MPEG4 Video Applications", In ICICT, 2009.

[77] Mao S, Panwar SS and Hou YT, "On Minimizing End-to-End Delay With Optimal Traffic Partitioning", IEEE Transactions on Vehicular Technology, 2009, 55(2); 681-690.

[78] Zhong F, Yeo CK and Lee BS, "Adaptive Load Balancing Algorithm for Multi-Homing Mobile Nodes in Local Domain", In Proceedings of IEEE CCNC, 2011.

[79] Manousakis K, Gopalakrishnan P, Famolari D and Van Den Berg E, "INTELiCON: Intelligent Connectivity Framework for the Simultaneous Use of Multiple Interfaces", In Proceedings of ICC, 2007.

[80] Koudouridis G P, Karimi H R and Dimou K, "Switched multi-radio transmission diversity in future access networks", In Proceedings of the IEEE Vehicular Conference, 2005.

[81] Kim J-O, Ueda T and Obana S, "MAC-level measurement based traffic distribution over IEEE 802.11 multi-radio networks", IEEE Transactions on Consumer Electronics, 2008, 54(3); 1185-1191.

[82] Kim J, "Feedback-Based Traffic Splitting for Wireless Terminals with Multi-Radio Devices", IEEE Transactions on Consumer Electronics, 2010, 56(2).

[83] Kim J-O, Davis P, Ueda T and Obana S, "Splitting downlink multimedia traffic over WiMAX and WiFi heterogeneous links based on airtime-balance", Wireless Communications and Mobile Computing. 2011.

[84] Schulzrinne H, Casner S, Frederick R and Jacobson V, "RTP: A transport protocol for real-time applications", RFC 3l550, 2003.

[85] https://www.nsnam.org/

[86] Gao C, Ling C, Z and Yuan Y, "Packet reordering analysis for concurrent multipath transfer", International Journal of Communication Systems 27.12 (2014): 4510-4526.

[87] Schulzrinne Henning, "Real time streaming protocol (RTSP)", (1998). RFC 2326

[88] Singh V, Karkkainen T, Ott J, Ahsan S and Eggert L, "Multipath RTP. Internet draft", IETF, 2011.

[89] Schwarz H, Detlev M and Thomas W, "Overview of the scalable video coding extension of the H. 264/AVC standard", Circuits and Systems for Video Technology, IEEE Transactions on 17.9 (2007): 1103-1120.

[90] Rosenberg J, "Interactive connectivity establishment (ICE): A protocol for network address translator (NAT) traversal for offer/answer protocols", No. RFC 5245. 2010.

[91] Jacobson V and Mark H, "SDP: session description protocol", 1998.

[92] Ott J, Wenger S, Sato N, Burmeister C and Rey J, "Extended RTP profile for real-time transport control protocol (RTCP)-based feedback (RTP/AVPF)", No. RFC 4585. 2006.'

[93] Wenger S, "H.264/AVC Over IP", IEEE Transactions on Circuits and Systems for Video Technology, 13.7 (2003): 645-656.

[94] Reichel, J., H. Schwarz and M. Wien. "Joint scalable video model 11 (JSVM 11)", Joint Video Team, Doc. JVT-X202 (2007).

[95] http://trace.eas.asu.edu/yuv/

[96] Vatolin D, Moskvin A, Petrov O and Trunichkin N, "Msu video quality measurement tool", http://www.download3k.com/Install-MSU-Video-Quality-Measurement-Tool.htrnl (2009). Accessed on February 2015

[97] Hore A and Djemel Z, "Image quality metrics: PSNR vs. SSIM", 20th IEEE Conference on Pattern Recognition (ICPR), 2010.

[98] Paasch C, Ferlin S, Alay O and Bonaventure O, "Experimental evaluation of multipath TCP schedulers", Proceedings of the 2014 ACM SIGCOMM workshop on Capacity sharing workshop. ACM, 2014.

[99] Wischik D, Raiciu C, Greenhalgh A and Handley M, "Design, Implementation and Evaluation of Congestion Control for Multipath TCP", NSDI. Vol. 11. 2011.

[100] Kang S R, Liu X, Dai M and Loguinov D, "Packet-pair bandwidth estimation: Stochastic analysis of a single congested node", Proceedings of the 12th IEEE International Conference on Network Protocols, 2004.

[101] Iyengar J R, Paul D Amer and Randall S, "Receive buffer blocking in concurrent multipath transfer", IEEE Global Telecommunications Conference, 2005.

[102] Kheirkhah M, Ian W and George P, "Multipath-TCP in ns-3", Workshop on ns-3 held in conjunction with SIMUTools 2011, Barcelona/Spain, Dec 2011.

[103] Ford A., Raiciu C, Handley M, Barre S and Iyengar J, "Architectural guidelines for multipath TCP development", RFC 6182. 2011.