PH.D. THESIS

Congestion Management and Medium Access Control in Satellite Data Networks

by Xiaoming Zhou

Advisor: Dr. John S. Baras

CSHCN PhD 2005-2
(ISR PhD 2005-2)

The Center for Satellite and Hybrid Communication Networks is a NASA-sponsored Commercial Space Center also supported by the Department of Defense (DOD), industry, the State of Maryland, the University of Maryland and the Institute for Systems Research. This document is a technical report in the CSHCN series originating at the University of Maryland.

Web site http://www.isr.umd.edu/CSHCN/

ABSTRACT

Title of Dissertation: Congestion Management and Medium Access Control in Satellite Data Networks

Xiaoming Zhou, Doctor of Philosophy, 2005

Dissertation directed by: Professor John. S. Baras
Department of Electrical and Computer Engineering

Satellites are now being used to carry Internet traffic which brings new opportunities as well as new challenges to the satellite industry. Our goal in this dissertation is to design efficient and fair transport layer and medium access control layer protocols for satellite networks. Three problems have been addressed in this dissertation.

The first problem is to improve TCP performance over satellite networks and a protocol called receiver window backpressure protocol (RWBP) is proposed to be used inside the satellite networks. RWBP has newly designed congestion control, error control and buffer management schemes. And it requires less reverse channel bandwidth without sacrificing the forward channel throughput. RWBP can still maintain good performance even when there is high priority traffic such as streaming video, audio competing with the TCP traffic in the satellite networks.

The second problem is to investigate the transfer of short messages with delay constraints in the reverse channel. A multichannel random access protocol called FMCSA is proposed in this dissertation. FMCSA combines random access with packet level forward error correction (FEC) coding for new messages and scheduled retransmissions for partially received messages. Through analysis and simulations, we show that FMCSA can achieve higher throughput and lower delay than slotted Aloha.

In the third problem, we conduct a capacity and performance study for interactive data services in wireless access networks. We evaluate the performance of several MAC protocols such as slotted Aloha, static TDMA, Aloha/periodic stream and combined free demand assignment multiple access (CFDAMA) using realistic web traffic models. Based on the performance evaluation, we propose a new MAC protocol. Our new MAC protocol called CPFDAMA explores the correlation between forward channel data packets and reverse channel acknowledgement packets. CPFDAMA combined with RWBP in the transport layer outperforms the proposed protocols in term of both channel utilization and page response time.

On the whole, we address the Internet access in satellite networks from a system perspective. Our work can be used to improve the performance of current systems and it also can be used to design next generation satellite networks.

Congestion Management and Medium Access Control in Satellite Data Networks

by

Xiaoming Zhou

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2005

Advisory Committee:

Professor John. S. Baras, Chairman/Advisor
Professor Gnanalingam Anandalingam
Professor Anthony Ephremides
Professor Richard J. La
Professor Armand M. Makowski

DEDICATION

To my grandmother and grandfather;

To my mother and father;

To my brother and

To my wife.

ACKNOWLEDGEMENTS

Pursuing a Ph.D. degree is a long and unique journey in my life. I would like to take this opportunity to express my gratitude to those who helped me through this journey.

First, I would like to thank my advisor, Professor John S. Baras. The decision he made in Spring 2000 to take me as one of his graduate assistants gave me the opportunity to do the research I am interested in. I am grateful to his support and advice ever since then. He creates a wonderful and dynamic environment for me to learn and gives me the freedom to explore the interesting problems.

I would like to thank Professor Gnanalingam Anandalingam, Professor Anthony Ephremides, Professor Richard J. La and Professor Armand M. Makowski for kindly serving on the committee and reviewing the dissertation.

Without Dr. Chunrong Ai's support, it's not possible for me to begin my study here at University of Maryland financially. I am grateful to his generosity. Thanks also go to Dr. Zhu Han who picked me up in Washington DC in an early morning 1999 and brought me to College Park physically. I am also grateful to Professor Victor Granatstein and Professor Jerome A. Gansman for generously allowing me to be a grader for their courses during my first semester so that the upper bound of the working hours was almost reached. I would like to thank Dr. Majid Raissi-Dehkordi and Dr. Jiashiang Jou for taking me back and forth to Germantown every single day during the summer internship with HNS. I am also grateful to Mr. Rod Ragland of HNS for his advice and support.

I am grateful to the financial support from National Aeronautics and Space Administration (NASA) under contract NCC8-235, NCC3-528, NAG59150 and from Hughes Network System through the Maryland Industrial Partnership program under contract HNS MIPS 2612.

I am grateful to the group members involved in the DARPA NMS project, Dr. Nelson Liu, Dr. Majid Raissi-Dehkordi, Dr. Jiashiang Jou, Huigang Chen, Karthikeyan Chandrashekar, Senni Perumal, George Papageorgiou, Gun Akkor, Ayan Roy-Chowdhury and Nalini Bharatula for many helpful discussions. Thanks also go to the HNS MIPS group members Manish Karir, Aniruddha Bhalekar and Do Byun. I would like to thank Dr. Michael Hadjitheodosiou for the discussions on satellite communications and students in his group Yadong Shang, Hui Zeng and Yingyong Chen for their help. Thank you all for teaching me so much.

Special thanks go to Althia Kirlew, Diane Hicks and Lisa Schuetz for their support and help.

I would also like to thank my friends Tianmin Ren, Chunxian Ye, Zhiyun Li, Jian Liang, He Huang, Tao Jiang, Xiaojun Li and all others. I also appreciate the help of Professor Babis Papadopoulos, Professor Mingyan Liu and Mr. Rod Ragland.

Finally, I would like to express my gratitude to my parents for their love and support anytime, anywhere on demand or without demand at all. I would like to thank my wife for her endless love and for coming here to experience the ups and downs with me.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

1.1 Internet over Satellite

Since the first communication satellite launched in 1957, satellite communications have experienced a rapid growth [40]. In the first twenty five years of commercial operation, the satellite capacity have increased 200 times. Earlier systems were primarily used as international voice trunks between large earth stations with 97 foot diameter antennas and costing millions of dollars. Earth stations have become smaller and cheaper ever since then. Now, Very Small Aperture Terminals (VSAT) with a diameter of less than 1.5 meters as well as handhold terminals are available. Recently satellites are used to carry Intenet traffic which brings new opportunities as well as new challenges for the satellite industry.

1.1.1 Satellite Communications

According to the orbital altitude, satellites can be classified as low earth orbit (LEO) satellites (400 to 1,000 miles in altitude), medium earth orbit (MEO) satellites (5,000 to 7,000 miles in altitude) and geo-synchronous orbit (GEO) satellites

(22,000 miles in altitude) [42] [15]. The lower altitude of LEO and MEO satellites allows smaller transmission power and easier support of delay sensitive applications. However since the LEO and MEO satellites are rotating relatively to the Earth, the system complexity is increased because of the need for satellite movement tracking and handover management. GEO satellites have the advantage of being stationary relatively to the Earth and being able to cover very large areas of the Earth. However the higher altitude of GEO satellites causes a larger propagation delay of about 125 ms and a less efficient link budget compared with LEO and MEO systems.

According to the processing capability, satellites can be classified as bent pipe satellites and on-board processing (OBP) satellites [4]. Bent pipe satellites are physical layer devices and they are simply signal repeaters in the sky. The signals received on the satellite uplink are amplified and broadcasted at a different frequency on the downlink. More advanced OBP satellites accommodate baseband digital signal processing, uplink bandwidth controller and fast packet switching on board. OBP satellites are link layer devices and they form a mesh network topology rather than a star topology as in the bent pipe satellite networks.

The Ku band (10 to 18 GHz) is now being used for satellite communications [15]. To support high-rate transmission, Ka band (27 to 40 GHz) is proposed for broadband applications. The main problem with Ka band is the significant rain attenuation which introduces much higher bit error rate compared with that in optical fiber networks. Satellite and terminal mobility combined with radio wave fading due to terrain (e.g. tunnels, mountains, foliage) also contribute to the satellite link errors.

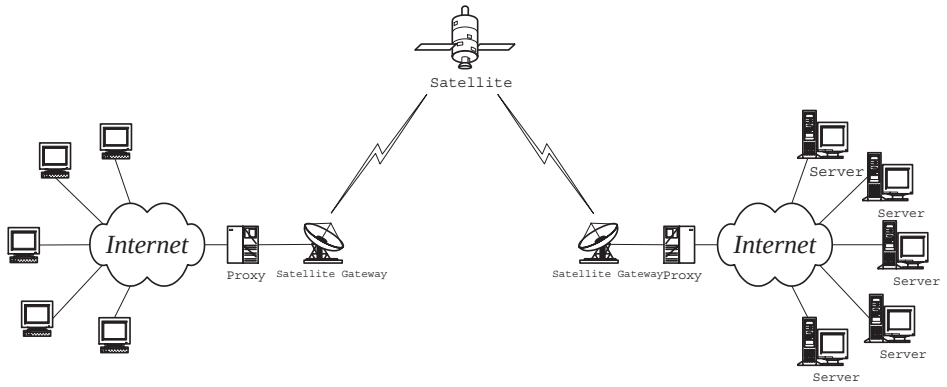


Figure 1.1: Bent pipe satellite used as a traffic trunk (point-to-point)

1.1.2 Satellite Located on the Internet: Core and Edge

With the competition of optical fiber and terrestrial wireless networks, we believe satellite networks will continue to be an indispensable part in the global information infrastructure. There are two basic properties which make satellite communications unique [40]: 1) ability to cover very large areas at any given instant of time; 2) satellite communication system is some form of wireless system. These two properties enable them to provide Internet services to both fixed and mobile users over wide areas including rural areas, water areas and large volume of air space. They also create the opportunity to allocate bandwidth on demand to a large number of users and to provide multicasting and unicasting communications with distance-insensitive costs.

As their application in the early days, satellites are continuing to serve as traffic trunks in the core Internet to connect two terrestrial networks which are far away from each other (figure 1.1). Satellite systems can also serve as access networks to provide Internet services directly to fixed and mobile users (figure 1.2).

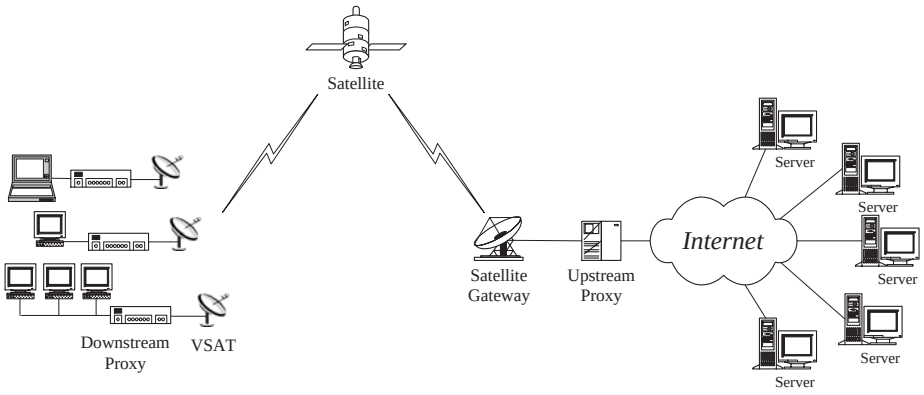


Figure 1.2: Direct to user bent pipe satellite network (star topology)

1.2 Statement of the Problems

In this dissertation, we are mainly interested in the GEO satellites. We focus on GEO satellites because they have the following advantages: stationary relatively to Earth, large covering areas and significant reduction in system complexity compared to LEO and MEO satellite systems. However, GEO satellites are about 22,000 miles above the Earth surface and introduce a propagation delay about 125 ms from the ground terminals to satellites.

In this dissertation, we focus on the integration of satellite networks with the terrestrial networks to provide data services, specifically TCP traffic. The data services are best effort services and no connection admission control (CAC) is enforced on this kind of traffic. As shown in figure figure 1.1 and 1.2, two interfaces are crucial to this integration. First is the terrestrial interface at the satellite gateways where the satellites are connected to the Internet. Second is the air interface at the satellite terminals where they are connected to the satellites.

Our goal is to design efficient and fair transport layer and medium access control (MAC) layer protocols for GEO satellite networks at the two interfaces. One

primary concern is to provide high speed Internet and Intranet access to home users and enterprise users, and allow the satellite communication companies to keep control of their networks. The pricing and performance of Internet and Intranet applications to large number of users should be competitive with cable and DSL services. The interesting and important problems about this concern we want to investigate are:

1.2.1 Congestion Management in the Transport Layer

Currently Internet depends on TCP at the end hosts and the tail-drop or random early drop (RED) [39] based queue management at the routers for congestion control. Because the long propagation delay and possibly high bandwidth in the satellite networks, it is quite possible that a lot of packets will enter the satellite networks and get dropped before the TCP sources start to decrease their sending rates. While resources, such as bandwidth, power and buffer, are extremely precious in the satellite networks. If the packets get dropped, all the resources they consumed are totally wasted. Therefore the current end-to-end architecture is not efficient and effective in satellite networks.

The problems we are targeting to solve are: What is the suitable congestion management architecture for satellite hybrid networks? How to improve the TCP performance in satellite networks, which have long propagation delay, large bandwidth-delay product, high bit error rate and forward/reverse channel bandwidth asymmetry? Furthermore the TCP traffic has to share the satellite bandwidth with multicasting UDP traffic such as streaming video and audio. Therefore another problem is how to maintain high satellite link utilization even with competing UDP traffic?

1.2.2 Medium Access Control in the Reverse Channel

In the star network as shown in figure 1.2, the traffic flowing in the reverse channel from the terminals to the hub mainly contains short messages such as HTTP requests and DNS lookups. A typical size of such messages is about 432 bytes [65] [23] [10] [86]. In the MAC layer, a short message is fragmented into multiple smaller MAC packets. For example, assume each MAC layer slot carries a MAC header of 5 bytes and a MAC payload of 48 bytes, therefore a short message of 432 bytes will be segmented into 9 MAC packets. Only after all the MAC packets of a message are received will the hub reassemble the message and forward it to the upper layer.

For short transaction based messages, it is desirable to deliver them with the smallest delay possible. For the multislot short messages we are interested in, the proposed random access protocols in the literature can deliver the messages with small delay only when they are operating at extremely low channel utilization region e.g. 1% in slotted Aloha. While reservation based protocols can achieve much higher utilization, the overhead delay introduced about 500 ms is too expensive for short messages. The problem we want to solve is how to transfer the multislot short messages in the reverse channels with small delay and reasonable utilization so that the satellite service provider can make profit.

1.2.3 Capacity Planning and End-to-end Performance

We are interested in capacity planning and performance evaluation for web browsing in satellite networks. The important metrics in web browsing over satellite access networks include page response time, average throughput and channel bandwidth utilization etc. We start to address this from the link level. We extend our

results from link level to end-to-end level by introducing a transport layer proxy at the boundary between the satellite and terrestrial networks.

The problems we are addressing include: 1) What is the realistic user traffic model for web browsing? 2) How much bandwidth is needed in the forward and reverse channel in order to provide certain quality of service to certain number of users? 3) What is the suitable MAC protocol for the bursty data traffic in the reverse channel? 4) How does the performance in satellite networks compare with that in terrestrial wireless networks with the same bandwidth? 5) What is the link level performance? 6) How to integrate the MAC protocol and the transport layer protocol to form an end-to-end congestion management mechanism? 7) What is the end-to-end performance for different transport and MAC layer protocols?

The objective of this dissertation is to integrate the current satellite networks with the terrestrial networks to provide efficient and fair Internet services. Rather than design the whole system from scratch, we adopt an evolutionary strategy. We will mainly focus on the star network topology shown in figure 1.2. The clients download web pages or large files from the Internet servers. The most important metrics to the users are the end-to-end throughput for large file transfers and page response time for short files. Because we cannot change the transport layer protocol on all the Internet servers, we introduce proxies between the satellite networks and terrestrial networks which are transparent to both the servers and clients. In the MAC layer, a fixed size slot is used. Therefore if the upper layer packet is larger than the slot size, the packet will be segmented before transmission. Because we are considering reliable communications, only after all the segments of an upper layer packet are received, they can be reassembled and be delivered to the upper layer. In the second problem, when we consider the random access

for short messages in the reverse channel. We assume the message length is 432 bytes which is the typical size of a page request. And we also assume the MAC slot size is 48 bytes. In addition to this constraint, the terminal also have the peak power constraint. Therefore when the reverse channel bandwidth is increase from narrowband to wideband, the terminals have to send with a higher power level in order to keep the energy per bit constant. Due to this peak power constraint, the wideband is divided into a number of sub-channels so that the peak power of each terminal is under the constraint. We are interested in the home and enterprise user and the terminals are AC powered rather than battery powered. Therefore a terminal can keep on sending as long as there are packets in its queue.

1.3 Contributions

In this dissertation, we have made the following contributions:

First, we proposed an edge-to-edge based congestion management architecture to be integrated with the end-to-end TCP congestion control and we developed a reliable transport layer protocol to be used inside satellite networks [97] [96] [99].

A satellite network forms a network domain and a proprietary protocol stacks are designed for it with its specific characteristics in mind. The terrestrial networks still use the standard TCP/IP protocol stacks. The satellite network is connected to the terrestrial networks by proxies and proxies do the protocol conversion between the two networks. The idea is to solve the local problem locally just because proxies have more information at the interfaces of the satellite and terrestrial networks and they know it earlier. Once the congestion is feedback to the proxies, it is back pressured to the end hosts.

Then a transport layer protocol called receiver window backpressure proto-

col (RWBP) is developed for the edge-to-edge congestion management inside the satellite networks. RWBP takes advantage of the specific characteristics of satellite networks. It has newly designed congestion control, error control and buffer management schemes. And it requires less reverse channel bandwidth without sacrificing the forward channel throughput. RWBP can still maintain good performance even when there is high priority traffic such as streaming video, audio competing with the TCP traffic in the satellite networks. It is seamlessly integrated with terrestrial TCP and the congestion inside the satellite network can be fed back to the Internet hosts.

Second, we developed a random access protocol for multislot short messages in the reverse channel [98].

In order to improve the delay performance of multislot short messages in a multiple access channel with long propagation delay, a multichannel random access protocol called fast multichannel slotted Aloha (FMCSA) is proposed in this dissertation. FMCSA combines random access with packet level forward error correction (FEC) coding for new messages and scheduled retransmissions for partially received messages. Through analysis and simulations, we show that FMCSA can achieve higher throughput and lower delay than slotted Aloha. When the system is operating at relatively low load region, the short messages can be delivered in their first attempts with very high probability. With the load increasing, more messages will be received partially in their first attempts and the scheduled retransmission scheme will guarantee the partially received messages to be recovered in their second attempts. We also show that the improved performance of FMCSA compared to slotted Aloha is robust to the FEC code rate, channel bandwidth, terminal population, arrival patterns, slot size as well as message length.

Third, we conducted a capacity and performance study for interactive data services in wireless access networks [100] [101].

In this study, the transmission time and the propagation delay in both forward and reverse channels are explicitly modeled. Therefore bottlenecks in both channels can be captured. We adopt a closed queuing model which is very general and can be applied to different wireless access works such as cellular networks, wireless metropolitan area networks (WMAN) and satellite networks. It can provide insight into the dynamics of the network and into the design of the MAC protocols in the reverse channel. From this model, we can calculate the important performance metrics such as the utilization in the forward channel, the file response time and average user throughput etc.

We evaluate the performance of several MAC protocols such as slotted Aloha, static TDMA, Aloha/periodic stream and combined free demand assignment multiple access (CFDAMA) using realistic web traffic models. Based on the performance evaluation, we propose a new MAC protocol and a new transport layer protocol. Our new MAC protocol called combined polling free demand assignment multiple access (CPFDAMA) explores the correlation between forward channel data packets and reverse channel acknowledgement packets. Our new transport layer protocol called RWBP uses per-flow queuing, round robin scheduling and receiver window backpressure for congestion management. RWBP can eliminate congestion losses inside the wireless networks. Our protocol suite outperforms the proposed protocols in term of both channel utilization and response time. Our results can be used for service providers to dimension their networks and provide quality of service to a certain number of users.

1.4 Dissertation Organization

The arrangement of this dissertation is as follows. In chapter 2, we give the performance metrics and design goals for congestion management and medium access control. The problems in the transport layer and the MAC layer are pointed out and the solutions proposed in the literature are reviewed; In chapter 3, we describe the congestion control, error control and buffer management in RWBP. Analytical and simulation results of RWBP are compared with TCP; In chapter 4, the random access protocol FMCSA is described and its throughput is analyzed. We also evaluate its delay performance with simulations. Chapter 5 discusses the capacity planning for both forward channel and reverse channel in order to provide certain quality of service to certain number of users for interactive data services. Then the end-to-end performance of web browsing is evaluated with a realistic traffic model and different combinations of transport and MAC layer protocols. Finally, chapter 6 concludes this dissertation.

Chapter 2

Background and Related Work

The problems we are addressing in this dissertation are essentially resource management schemes at two different layers. For the transport layer problem, the management scheme is a distributed algorithm running at the end-hosts and possibly coordinates with buffer management mechanism at the routers. The reverse channel multiple access control is a many-to-one problem and the algorithm could be distributed or centralized. In order to design a better protocol, it is necessary to understand the requirements and performance metrics of a resource management scheme.

2.1 Performance Metrics and Design Goals

Efficiency: Efficiency is the portion of the channel capacity used for data transmission [41]. The resource management scheme should take advantage of statistical multiplexing and be able to match the aggregate traffic load with the channel capacity.

Delay: Delay is the time for a packet or a message to reach the destination.

Long message applications, such as FTP and email, are more sensitive to average throughput than to single packet delay. While for transaction based applications such as HTTP, telnet and remote login, single packet delay is more important because the message is short and message delay cannot be amortized as in long message applications.

Fairness: Fairness deals with allocating the resources among connections and does not exhibit preference to any connection [41]. This definition can be modified when traffic with different weights is handled and fairness controller should allocate resources proportional to their weights.

Scalability: Internet is an open system and evolves rapidly. The resource management scheme should be scalable in terms of link capacity, propagation delay, number of flows and number of users. Scalability is a key property in satellite networks to support hundreds of thousands of users with a single transponder.

Convergence: Given sustained source loads and network configuration, the scheme should bring the network to a stable state [51]. In an unstable network, the total load would either increase to infinity or decrease to zero with time. When network configuration and traffic vary, the scheme should respond quickly and converge to a new stable state. Once operating at the stable state, it is desirable to have smaller amplitude of oscillation around the equilibrium point.

Robustness: Internet connects a wide range of hardware and software systems and the resource management scheme should be able to cope with the heterogeneity. Furthermore it should be robust to parameter setting, control packet loss, channel error, malicious or misbehaved users, different packet sizes and different services.

Feasibility: The resource management scheme should be simple to be specified

and simple to be implemented. And the implementation and operation cost should be cheap even for large number of users.

Usually it is difficult to design a scheme which satisfies all the above properties and some are even conflicting with each other. For example, high efficiency usually means large delay. Simple distributed scheme usually means low efficiency and instability. Therefore we have to deal with the tradeoffs and come up with a scheme which fits best in the traffic arrival pattern and satisfies the quality of service.

2.2 Congestion Management in Satellite Networks

2.2.1 TCP Overview

TCP is a connection-oriented, end-to-end reliable transport protocol. TCP source and destination port number combined with IP source and destination addresses uniquely identify each TCP connection. There are three important mechanisms in TCP i.e. flow control, congestion control and error control [78] [87] [50] [64].

TCP Flow Control

TCP uses a sliding window to achieve flow control. The TCP receiver sets the receive window field (RWND) in the acknowledgement to its free buffer size so that the sender will never overflow the receiver's buffer. Therefore the receiver buffer size sets an upper bound of the in flight packets of the connection. Our proposed solution in chapter 3 uses this mechanism to push the congestion back to the source by the reduced receiver window.

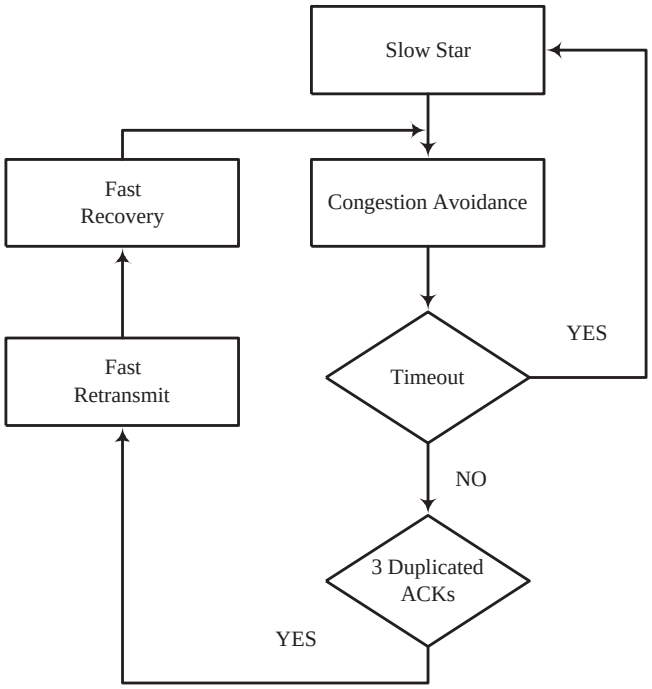


Figure 2.1: TCP congestion control: slow star, congestion avoidance, fast retransmit and fast recovery

TCP Congestion Control

The TCP sender maintains a state variable CWND for congestion window size. While RWND is used to guard that the sender will not overflow the receiver buffer, the CWND is used to guard the sender will not overload the network. The TCP sender can send at most the minimum of RWND and CWND window worth packets without receiving any acknowledgement.

In the most popular version of TCP i.e. TCP Reno, there are four algorithms used for congestion control which are slow start, congestion avoidance, fast retransmit and fast recovery as shown in figure 2.1. Slow start is used upon the start of a new connection to probe the network bandwidth, it increases the CWND by one maximum segment size (MSS) when an acknowledgement is received, which

results in increasing congestion window exponentially. TCP stays in slow start until its CWND is greater than the slow start threshold. After that TCP gets into congestion avoidance, it increases CWND about one MSS per round trip time (RTT). Fast retransmit algorithm is triggered when a fixed number of duplicate acknowledgements (usually three) are received. TCP retransmits the potentially lost packet indicated by the acknowledgement and cuts its CWND to half. After that, it inflates its CWND by one MSS when a duplicate acknowledgement is received. If there is one and only one packet lost in a single window, the inflation can increase the CWND to the original CWND before the loss after about half RTT. After that TCP can send a new packet when each duplicate acknowledgement is received if allowed by the RWND. Finally it will send half a window new packets when it receives the first non-duplicate acknowledgement. TCP Reno is an advanced version of TCP Tahoe, which does not have the fast recovery algorithm and sends half a window packets in burst after the loss has been recovered.

TCP Error Control

Error control is the main component of any reliable protocol, which includes error detection and error recovery. TCP uses acknowledgement packet, timer and retransmission to achieve error control. TCP uses cumulative acknowledgement, which means when a packet gets lost, it prevents the acknowledgement from being advanced and the window cannot slide until the lost packet is recovered. The sliding window mechanism actually ties the flow control, congestion control and error control together and it becomes vulnerable when there are losses including congestion loss and packet corruption in the network [25].

2.2.2 Problems of TCP in Satellite Networks

Long Propagation Delay

A geo-synchronous orbit (GEO) satellite is about 22,000 miles above the earth. The propagation delay between the ground terminal and the satellite is about 125ms. Therefore a typical round trip time (RTT) in the satellite network is more than 500ms.

The time taken by TCP slow start to reach the satellite bandwidth (SatBW) is about RTT*log_2 (SatBW*RTT) when every TCP segment is acknowledged [6] [77]. For a connection with large RTT, it spends a long time in slow start before reaching the available bandwidth. For short transfers, they could be finished in slow start, which obviously does not use the bandwidth efficiently. Some researchers propose to use a larger initial window [7] up to about 4K bytes rather than one maximum segment size (MSS) for slow start. So files less than 4K bytes can finish their transfers in one RTT rather than two or three RTTs. Another proposal [35] is to cancel the delayed acknowledgement mechanism in the slow start so every packet is acknowledged and the sender can increase its congestion window (CWND) more quickly.

For bulk transfers, TCP throughput is inverse proportional to RTT [75]. So TCP connections with larger RTTs do not get their fair share of the bandwidth when they compete with the connections with smaller RTTs. Using simulations, Henderson claims the 'constant-rate' additive increase policy can correct the bias against connections with long RTTs [43]. However it is difficult to implement this policy in a heterogeneous network.

Large Bandwidth Delay Product

The bandwidth delay product in the satellite hybrid network is very large. In order to keep the large pipe full, the window should be at least the bandwidth delay product. However, the receiver advertised window that is 16 bits in the TCP header cannot be more than 64k bytes, which limits the two-way system throughput to 64k/580ms i.e. 903Kbps. Window scaling [49] is proposed to solve this problem. But when the window is large, it is more likely that multiple packets are lost in one window caused either by congestion or link layer corruption or both. The multiple losses will trigger TCP congestion control algorithms and lead TCP to actually operate with a small average window.

For the same reason, the sender buffer size can also limit the TCP connection throughput if it is less than the bandwidth delay product, which is usually the case in a lot of operating systems.

Channel Error

Satellite channel is noisier than optical fiber channel. Bit error rates (BER) of the order of 10^{-6} are often observed [44]. For mobile users, the bit error rates can be even higher because of channel fading and mobility. Because TCP Reno treats all losses as congestion in the network, this kind of link layer corruption can cause TCP to drop its window to a small size and lead to poor performance.

TCP SACK [66] can convey information about non-contiguous segments received by the receiver in the acknowledgements (ACKs) so that the sender can recover errors much faster than TCP Reno, which well known can recover only one loss per RTT. Forward error correction (FEC) coding is usually used in satellite communication to reduce the bit error rate. However, FEC consumes addi-

tional bandwidth [14] by sending redundant information together with the data and transforms the original random error nature to one with bursty errors.

Reverse Channel Congestion

For the star topology in figure 1.2, the forward channel bandwidth from the satellite to the earth terminals is much larger than the reverse channel bandwidth [8]. When the reverse channel traffic load is greater than the its bandwidth, congestion could happen. The congestion may cause poor performance in the forward because TCP uses ACKs to clock out data. In the best case, the ACKs are not lost, but queued, waiting for available bandwidth. This has a direct consequence on the retransmission timer and slows down the dynamics of TCP window.

In one way transfer, most of the time the reverse channel is used to transfer pure ACKs. To alleviate this problem, ACK filtering [12] was proposed to drop the ACKs in the front of the IP queue by taking advantage of the cumulative acknowledgement strategy in TCP. The situation is even worse for two-way transfers. When the users are sending data (say email with a large attachment or upload file using FTP) and browsing the web at the same time, a lot of data packets could be queued in front of ACKs in a FIFO queue, which increases the ACKs delay dramatically. In this case, a priority queue can be used to schedule the ACK to be sent first [12].

For the topology in figure 1.1 in which the satellite is used as a traffic trunk, the forward and reverse channel bandwidth usually is symmetric. However it is still desirable to consume as less bandwidth as possible to transfer acknowledgements without degrading the data channel performance.

Competing with UDP traffic

Usually the satellite service provider provides not only Internet services but also other services such as audio, video multicasting using the same satellite link [99]. Because of the higher revenue brought by this kind of traffic, it is given a higher priority. The leftover bandwidth is used by the Internet traffic. The increasing arriving rate of the high priority traffic could cause TCP packets get dropped and lead to low efficiency. To maintain high utilization of the satellite link, the Internet traffic has to adapt its sending rate to fill in the leftover bandwidth.

2.2.3 End Host Based Approach

There are several end host based approaches which try to improve the TCP performance. These end host based schemes fall into three categories: 1) window based such as TCP Vegas, TCP Westwood and TCP Peach; 2) rate based such as NETBLT; 3) hybrid approach such as SCPS-TP.

Window Based Solutions

TCP Vegas: TCP Vegas [19] addresses the congestion control problem from another perspective. It uses the transmission rate rather than packet loss as a congestion signal. Every round trip time, the sender calculates its transmission rate based on the sending window and the measured RTT. This rate is compared with the expected rate, which is sending window divided by Base RTT. Base RTT is the smallest RTT measured so far. The basic idea is that if there is no congestion in the network the measured rate should be closed to the expected rate. A low and a high thresholds are used to trigger window additive increase or decrease, depending of whether the channel is under-utilized or over-utilized. TCP Reno

always increases its congestion window if there is no loss and periodically causes the packet dropped and window oscillation. TCP Vegas can decrease its window in congestion avoidance, it reduces window oscillation dramatically once it reaches the equilibrium.

However, in TCP Vegas each connection may keep some packets in the bottleneck router buffer, therefore TCP Vegas is not stable when the number of connections becomes large. Furthermore, the RTT measured by the sender may be caused by the congestion in the reverse channel rather than the forward channel. So TCP Vegas does not work well in the asymmetric channel case.

TCP Westwood: TCP Westwood [21] is a sender-side modification of the TCP fast recovery algorithm that improves upon the performance of TCP Reno in wireless and satellite networks. TCP Westwood estimates the bandwidth used by the connection by monitoring the inter-arrival time of the returning ACKs. This estimate is then used to compute congestion window and slow start threshold after congestion is detected either by three duplicate ACKs or by timeout. TCP Westwood attempts to choose a slow start threshold and a congestion window in consistence with the effective bandwidth when congestion is experienced. While under same circumstance, TCP Reno blindly halves the congestion window.

TCP Westwood outperforms TCP Reno particularly when there are channel introduced errors such as in wireless and satellite networks. However, as in TCP Vegas, the rate of returning ACKs depends on the reverse channel access scheme and congestion status. Therefore the inter-arrival time may not reflect the effect bandwidth in the forward channel.

TCP Peach: TCP Peach [6] has two new algorithms sudden start and rapid recovery, which replace the slow start and fast recovery algorithms in TCP Reno

respectively. Essentially TCP Peach has two logical channels, one is for the data transmission and another one is for bandwidth probing. TCP Peach uses low priority dummy segments to probe the bandwidth in sudden start and rapid recovery. The problem with TCP Peach is that dummy segments do not carry any information and they are overhead to the data. Another problem is that all the routers need to implement some kind of priority mechanism, which makes it difficult to deploy.

Rate Based Solutions

NETBLT: NETBLT [25] is a reliable transport layer protocol designed for bulk data transfer over satellite IP networks. NETBLT is motivated by the following two observations. First, window and timers perform poorly in synchronizing the end states. Second, widow mechanism ties flow control and error control together, therefore becomes vulnerable in the face of data loss. NETBLT uses rate rather window for flow control and the rate control parameters are negotiated during the connection setup and periodically updated. And flow control and error control are decoupled in NETBLT.

Combined window rate based solutions

SCPS-TP: Space communication protocol standards-transport protocol (SCPS-TP) [31] is a set of TCP extensions for space communications. This protocol adopts the timestamps and window scaling options in RFC1323 [49]. It also uses TCP Vegas low-loss congestion avoidance mechanism.

SCPS-TP receiver doesn't acknowledge every data packet. Acknowledgements are sent periodically based on the RTT. The traffic demand for the reverse channel

is much lower than in the traditional TCP. However it is difficult to determine the optimal acknowledgement rate and the receiver may not respond properly to congestion in the reverse channel. It does not use acknowledgements to clock out the data rather it uses an open-loop rate control mechanism to meter out data smoothly. SCPS-TP uses selective negative acknowledgement (SNACK) for error recovery. SNACK is a negative acknowledgement and it can specify a large number of holes in a bit-efficient manner.

2.2.4 Network Assisted Approach

Network layer active queue management (AQM) such as RED, BLUE and FRED can improve TCP throughput and fairness by smart dropping or marking of IP packets at the bottleneck routers. In wired line networks losses are mainly caused by congestion, AQM could be particularly effective. However in wireless and satellite networks besides congestion losses, channel errors are common. Two strategies have been proposed to shield the channel errors from sender side TCP so that TCP will not misinterpret them as congestion. One strategy as in SNOOP and AIRMAIL is to recover the errors at the link layer. Another strategy is to split the end-to-end TCP connection at the wired/wireless network gateway. I-TCP, M-TCP, Super TCP and STP adopts this strategy. TCP connection splitting can not only shield the channel errors from the source TCP, but also shield the possibly long and variable delay from the source. A more revolutionary approach XCP [52] requires modifications to both end-host protocol and network router software.

RED: RED [39] gateway monitors the average queue size with a low pass filter. The decision to accept an incoming packet is based on whether the average queue size exceeds or conforms to predefined thresholds. If the average queue size is below the low threshold, the arriving packet is accepted. If the average queue size is between the low and high threshold, the arriving packet is dropped or marked with probability proportional to the average queue size. If the average queue size exceeds the high threshold, the arriving packet is dropped or marked with probability one. Some form of per flow fairness is ensured in RED since the packet drops of a specific connection during congestion is roughly proportional to its arrival rate. RED attempts to maintain a low average queue size while admitting occasional bursts of packets. However, the random drop of incoming packet in RED could cause the TCP source congestion window to be halved even if the connection does not exceed its fair share. Therefore RED may suffer from short term unfairness.

BLUE: Based on the observation that the packet loss rate is still high even with RED, BLUE [36] uses packet loss and link idle events rather than queue length to manage congestion. The packet marking probability is increased when there are packet drops due to buffer overflow; conversely when the queue is empty or the link is idle, the marking probability is decreased. By decoupling congestion management from instantaneous or average queue length, BLUE has been shown to perform significantly better than RED both in term of packet loss and buffer size requirements in the routers.

FRED: Flow random early detection (FRED) [62] was motivated by the unfair bandwidth sharing of RED gateways between adaptive and non-adaptive flows.

This is because RED imposes the same loss rate on all the flows regardless of their bandwidth. FRED maintains per-active-flow accounting and the drop probability is proportional to the flow's buffer usage. FRED provides greater fairness, flow isolation and flow protection than RED.

Link Layer Error Recovery

SNOOP: Snoop TCP [13] was originally designed for last-hop wireless network. Snoop essentially uses the TCP acknowledgements to trigger the link layer retransmission at the base station and suppresses the duplicate ACKs from propagating to the TCP sender. Therefore it can shield the channel errors and does not drive the TCP sender to cut its window to half as end to end TCP does. Although Snoop does not have any TCP layer code running at the base station, it still need to access the TCP header to get the sequence number and acknowledgement number. It does not work if IPSEC is used. Snoop preserves the end-to-end semantics of TCP. However Snoop cannot be used for satellite network because the long propagation delay of the satellite link could cause fairness problem if the base station keeps the ACKs to transmit end to end.

AIRMAIL: AIRMAIL [9] is a reliable link layer protocol developed for indoor and outdoor wireless networks. By combining link level ARQ and adaptive Forward Error Correction (FEC) coding, it recovers the channel errors locally and can obtain better throughput and latency performance. However there exists a complex interaction between the reliable link layer and end-to-end TCP. It is possible for the error to trigger the link layer retransmission while at the same time the duplicate ACKs of TCP propagate to the TCP source and cause TCP to halve its window and to retransmit the same packet. Another problem is that not all the up layers

need reliable link layer service, e.g. real-time traffic using UDP does not need reliable data transmission.

TCP Connection Splitting

I-TCP: The basic idea of indirect TCP [11] is that the end-to-end TCP connection is divided into two connections, one is from the server to the base station and another one is from base station to the mobile users. The base station sends premature acknowledgements to the server and takes responsibility to relay the data to the mobile host reliably. The advantages are separation of flow control and congestion control of wireless and wired network, faster reaction to channel errors. However this scheme violates the end-to-end semantics of TCP. In I-TCP, it is possible the sender receives an acknowledgement of a data packet while the data packet has not reached the destination rather is buffered at the base station. The authors argue that many applications such as FTP and HTTP use application layer acknowledgements in addition to end-to-end TCP acknowledgements. Using I-TCP for these applications does not comprise end-to-end reliability.

M-TCP: M-TCP [20] also uses connection splitting. One novel idea in M-TCP is that the base station may set the advertised receiver window size to zero so that it can choke the sender and force it into persist mode. While in persist state, the TCP sender will not suffer from timeouts, retransmission timer back off and the congestion window stays open. This idea can be used as a flow control mechanism by the base station when there is congestion at the base station. TCP Connection splitting is always criticized for extra processing overhead and scalability problem. Using experiment, this paper shows that a 100 MHz Pentium PC can adequately handle numerous simultaneous connections with little performance degradation.

Super TCP: Because GEO satellite channel is a FIFO channel, there is no out-of-order routing. And congestion over the satellite link is impossible if the packets are sent at the rate of the satellite bandwidth. In [70], a connection splitting based solution is proposed to use one duplicate ACK to trigger the fast retransmission at the upstream proxy and to use a fixed window size for the satellite TCP connection. If there is only one connection in the system, the fixed window can be set to the satellite bandwidth delay product. However multiple connections with different terrestrial round trip times and different traffic arrival patterns have not been addressed. The paper proposes a new sender algorithm using the same idea as in TCP new Reno [38] [46]. It uses partial ACKs to calculate the bursty loss gap and sends all the potentially lost packets beginning from the partial acknowledgement number. Although it is possible that the sender could retransmit packets that have already been correctly received by the receiver, it is shown that this algorithm performs better than TCP SACK in recovering bursty errors.

STP: Satellite transport protocol (STP) [44] adapts an ATM-based protocol for use as a transport protocol in satellite data networks. STP can get comparable performance to TCP SACK in the forward channel with significantly less bandwidth requirement in the reverse channel. The transmitter sends POLL packets periodically to the receiver, the receiver sends STAT packet as acknowledgements. Therefore the reverse channel bandwidth requirement depends mainly on the polling period, not on the forward channel data transmission rate. So the bandwidth demand for the reverse channel decreases dramatically. STP uses a modified version of TCP slow start and congestion avoidance algorithms for its congestion control.

2.2.5 Combined Approach

XCP: Explicit control protocol (XCP) [52] is a more revolutionary approach which requires modifications to both end-host protocol and network router software. XCP generalizes the explicit congestion notification (ECN) [37] and the Core-Stateless Fair Queuing (CSFQ) [88]. More control information including sender's current congestion window, sender's round trip time estimate and a window feedback field is conveyed in the packet header. XCP is a window based congestion control protocol designed for best effort traffic in high bandwidth-delay product networks specifically optical fiber networks. XCP decouples the efficiency and fairness control. The efficiency controller uses Multiplicative-Increase Multiplicative-Decrease (MIMD) law and the fairness controller uses Additive-Increase and Multiplicative-Decrease (AIMD) law. It is shown by simulations that XCP maintains good utilization and fairness, has very shadow queue and drops very few packets.

The implicit assumption in XCP is that the link capacity is fixed and known. Although this is true for point-to-point links, it is not true for multiple access links whose capacity depends on the channel conditions and traffic characteristics of all nodes connected. It is not known whether XCP will work for multiple access links.

2.3 Medium Access Control in the Reverse Channel

2.3.1 Medium Access Control Overview

In the star satellite network as figure 1.2 shows, the reverse channel bandwidth is shared by all the terminals in the same transponder. Therefore the reverse channel

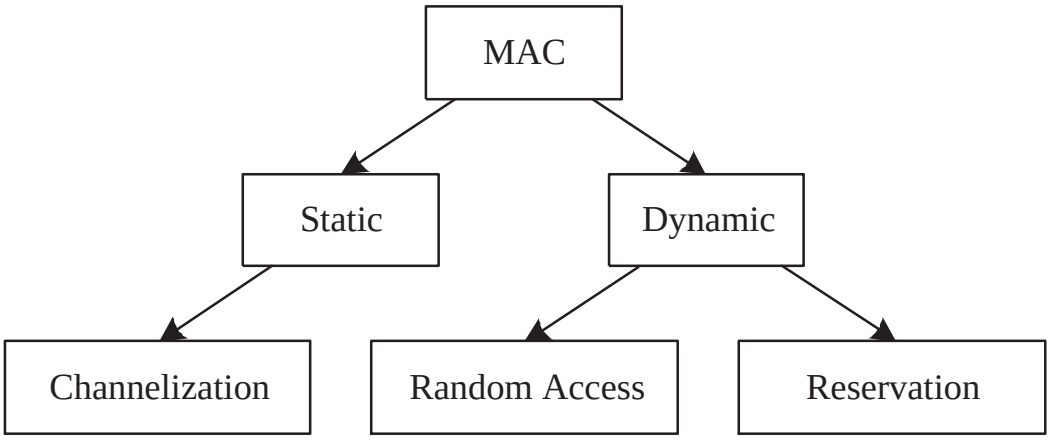


Figure 2.2: Classification of MAC schemes

is a multiple access channel (figure 1.2) and a MAC scheme is needed to allocate its bandwidth efficiently so that it will not become the bottleneck of the end-to-end performance.

According to the control policy, MAC schemes can be classified into static or dynamic MAC [61] (figure 2.2). Channelization falls into the static MAC and it includes TDMA, FDMA and CDMA. A terminal is assigned a fixed amount of bandwidth i.e. a fixed number of slots in a frame, a dedicated frequency or a dedicated code. The bandwidth is permanently assigned to a terminal for its lifetime [47]. The channelization is suitable for the gateways with relatively smooth aggregate traffic arrival patterns (figure 1.1). For the VSAT with very bursty traffic load (figure 1.2), the efficiency of channelization is low.

Dynamic MAC includes random access and reservation-based MAC. In random access, different terminals may send packet simultaneously and cause collisions. After the terminal detects the collision, it backs off for a random period of time and retransmits the lost packet. Pure Aloha and slotted Aloha belong to random access. Random access obtains reasonable throughput only at low load and offers

little quality of service guarantees such as delay, jitter etc.

Reservation-based MAC schemes are also known as demand assignment multiple access (DAMA). The basic idea of reservation-based scheme is to organize the distributed queues at the terminals into a coordinated global queue. In a centralized reservation-based scheme, the terminals send requests to a central controller located either at the network operation center (NOC) or on-board at the satellite. A terminal can either request total number of slots required to empty its queue (volume based) or it can request a variable number of slots in each frame and update its request periodically (rate based). The requests can be sent through a fixed assigned TDMA channel or through a random access channel. For the scenario we are interested in, there could be hundreds of thousands of terminals in a single transponder. Assigning each terminal a separate request channel is not efficient and random access should be used for the request channel [59].

In order to decrease the collision probability in the request channel, a request could be sent through the random access channel only when a burst arrives at an empty MAC layer queue, otherwise requests can be piggybacked on the data packets. Essentially DAMA shifts the collisions from the data channel to the request channel. Because the request packet size is much smaller than the data packet size, the slots wasted by idles and collisions are all short, leading to a higher overall efficiency [17].

The disadvantage of DAMA is that the reservation phase takes a significant amount of time in the satellite network because of the large propagation delay. This could increase the packet delay by two hops (about 0.5 sec) if the bandwidth controller is located at the NOC or by one hop (0.25 sec) if the bandwidth controller is located on board at the satellite.

2.3.2 Problems of Web Browsing in the Reverse Channel

Web browsing is characterized by the clients send small requests and the servers reply with small files. The metrics, which are important to web users, are the object response time, the web page response time and their variance. The object response time is the time elapsed between the time its request is sent and the time the object is received by the client. The web page response time is the time elapsed between the beginning of the connection establishment and the time of the last object finishes its transfer.

In this dissertation, we are only interested in the delay caused by the network. We do not model the server's service time which means that the response is sent to the client as soon as it is received by the server. The metrics, which are important to the satellite network service providers, are satellite link utilization, number of users supported, average queue size and packet drop probability at the satellite gateways and at the VSATs.

Problems in TCP for Short Web File Transfer

There are three aspects in the transport layer, specifically in TCP, that increase the end-to-end web response time dramatically:

First is the slow start algorithm in TCP congestion control. When a TCP connection is set up, it first enters the slow start phase with the initial congestion window size set to one segment. The idea for slow start is to probe the network bandwidth so that TCP can operate in an equilibrium state in the congestion avoidance phase. Since most of the web file sizes are small, they usually complete the transmissions in the slow start phase before they could enter the congestion avoidance phase. Hence, the bandwidth is not utilized efficiently for delivering

small web files and the response time will increase dramatically due to the long propagation delay in the satellite networks.

Second is the error recovery schemes in TCP. Packet could be lost due to congestion or channel error. One packet loss in slow start phase leads to a very small congestion window and low throughput. Two or more packet losses in slow start may cause TCP timeout and even connection reset. Note that TCP detects packet loss by three duplicate acknowledgements. For a small web file, the congestion window may be too small so that it could not have enough data packets to trigger the receiver to generate three duplicate acknowledgements. If packets cannot be recovered by this fast recovery scheme in TCP, TCP has to depend on its retransmission timer for error recovery. The default value of the retransmission timer is usually large and the end-to-end delay could be increased dramatically by timeout events.

Third is the FIFO queuing scheme in current Internet. This FIFO policy could cause three problems: 1) a bulk data transfer such as FTP download could have a lot of packets buffered in the FIFO queue at the satellite gateway, which increases the packet queuing delay of short files; 2) the large number of bulk data packets in the FIFO queue could cause buffer overflow and the packets of short files could be dropped due to congestion. This congestion loss will again trigger the inefficient error recovery schemes for short files and lead to an increased end-to-end object response time; 3) The situation is even worse for two-way transfers. When the users are sending data (say email with a large attachment or upload file using FTP) and browsing the web at the same time, a lot of data packets could be queued in front of ACKs in a FIFO queue, which increases the ACKs delay dramatically.

Incompatibility between TCP and MAC protocols

TCP is a self-clocking based protocol. A TCP receiver sends an acknowledgement every other segment and a TCP sender increases its congestion window based on the number of acknowledgements received. If the TCP ACKs are delayed in the network or when the number of ACKs received by the TCP sender is decreased as in ACK filtering [12], it will slow down the increase of the TCP sender's congestion window. Therefore the forward channel performance in term of object response time is proportional to acknowledgement delay and inverse proportional to acknowledgement frequency (equation 2.1). For web browsing, a client sends HTTP requests to download web pages from Internet servers. The HTTP requests delay will also contribute to the response times experienced by the VSAT web users. The request delay includes the queuing delay at the VSATs and the propagation delay. Equation 2.1 also shows the object response time is proportional to the delay of the HTTP requests.

$$Object\ response\ time\ \propto\ \frac{(TCP\ ack\ delay,\ HTTP\ request\ delay)}{TCP\ ack\ frequency} \tag{2.1}$$

$$Avg\ number\ of\ users\ supported = BW * \frac{BW\ efficiency\ of\ MAC}{Avg\ load\ per\ user} \tag{2.2}$$

From equation 2.2 we can see in order to increase the number of users the system can support provided that the total reverse channel bandwidth (BW) is fixed, we can decrease the average load per user by sending less frequent acknowledgements. However less frequent acknowledgments slow down the TCP sender congestion window dynamics and therefore increase the object response time (equation 2.1).

Another way to increase the number of users supported is to increase the reverse

channel bandwidth efficiency. The ideal MAC scheme would have both a high bandwidth efficiency and low packet delay. However such an ideal MAC scheme does not exist and high efficiency usually means large MAC delay. Large MAC layer delay will cause large ACK delay and large request delay. And it will eventually increases the web response time (equation 2.1).

2.3.3 Channelization

There are several proposed solutions in the literature about improving web browsing performance over satellite networks. Some solutions [44] [6] [76] either assume the reverse channel is a point-to-point satellite link or a dedicated telephone line. And the reverse channel multiple access problem has not been explored. We classify these solutions as channelization MAC schemes.

Henderson [44] does experiments based on the traffic generated according the HTTP trace distributions in [65]. In his experiments, Henderson assumes no packet loss and fixed RTTs so the results he gets are the latencies for the best case. He investigates both end-to-end TCP and connection splitting based schemes. However, in his experiments the satellite channel is abstracted as a link with large delay and the multiple access in reverse channel hadn't been been addressed.

In TCP Peach [6], a file of fifty segments is used to represent a web page and it is transferred consecutively from a server to a client. TCP Peach is shown to perform better than TCP Reno in term of average transfer time of web pages. The traffic trunk topology as in figure 1.1 is used for the simulations. TCP Peach over a multi-access channel hadn't been tested and the traffic model for web access is oversimplified.

Based on the observation that TCP slow start yields poor performance for

short and bursty web traffic in term of bandwidth utilization and page response time, Padmanabhan [76] proposes a fast start algorithm to replace the slow start algorithm in TCP. In fast start, the sender caches network parameters so that it can avoid paying slow start penalty for each web page. However when the cached parameters are stale, fast start could send packets too aggressively and has to recover the lost packets. The reverse channel for the satellite network simulations was a dedicated telephone line.

2.3.4 Random Access

Because the reverse channel traffic mainly composes of very bursty short messages i.e. TCP ACKs and HTTP requests, random access protocols are proposed to improve the delay performance.

Spread Aloha [2] is an wide band version of Aloha. It can provides connection free multiple access capabilities for large number of terminals. A common code is used for all the terminals and the terminals are separated by a random timing mechanism as in conventional Aloha. Spread Aloha can increase the instant bandwidth of each terminal, decrease the packet delay and satellite dish size. However it still suffers the low efficiency and instability problem as in narrow band Aloha.

Selective-Reject Aloha (SREJ-Aloha) [80] [81] is an unslotted random access protocol. It involves a subpacketization of messages in conjunction with a selective reject ARQ retransmission strategy. Because long messages usually overlap with each other partially, retransmitting those overlapped subpackets rather than the whole long message could improve the channel efficiency. It has been shown that SREJ-Aloha typically achieves a maximum throughput in the region of 0.4-0.45 along with excellent stability and delay properties.

2.3.5 Reservation

To improve the reverse channel efficiency, reservation based schemes are proposed. These schemes are exploring the specific characteristics of TCP traffic and trying to operate at a desirable efficiency delay region.

Connors [26] proposes a response initiated multiple access (RIMA) scheme for web access over a star satellite network. The basic idea is based on the following observation: when a large packet is received, with high probability the receiver will send an acknowledgement; when a small packet is received, with reasonable probability the receiver will send a data packet. The satellite acts as a scheduler. It checks the packet size and port number to find out how many slots needed by the receiver and try to allocate enough slots to the receiver. Although on-board switching is assumed, usually the satellite cannot check the transport layer port number because it is a layer two device. Therefore this scheme is expensive in practice. Connors runs end-to-end TCP for his simulations and the TCP over satellite problems have not been addressed.

Choi [34] developed a web traffic model. He uses his web traffic model and normal TCP in his simulations. The MAC protocol he uses in the simulation is adopted from a previous research in the context of hybrid fiber coax (HFC) networks. By grouping and piggy backing MAC requests, the MAC protocol can mitigate the performance degradation caused by large propagation delay. He also shows that the MAC protocol performs better than slotted Aloha.

Mhatre [69] considers the problem of web browsing over a multiple access channel. He proposes to use connection splitting for the bent pipe satellite networks. STP [44] is used for satellite connections and TCP is still used for terrestrial connections. Because the delay by having each and every packet to request a time

slot from the NOC is too long in DAMA, Mhatre proposes a scheduling algorithm which combines DAMA and free assignment. In his simulations, the clients always request objects of fixed size 4KB. He over-provisions the downlink bandwidth and the congestion over the downlink at the gateway hasn't been addressed.

2.4 Summary

For congestion management in the transport layer, TCP Vegas, TCP Westwood and TCP Peach need modifications at all the TCP sources which makes them difficult to deploy. The active queue management schemes are generic schemes to solve the TCP congestion control problem and are not tailored for satellite networks. The bandwidth sharing between TCP and UDP traffic has never been addressed in the related work. Usually what is assumed is that the satellite link is used exclusively by TCP traffic.

We will follow the TCP connection splitting approach and design a new transport layer protocol called receiver window backpressure protocol (RWBP) which takes all the problems listed in section 2.2.2 into account and RWBP is described in chapter 3.

For medium access control in the reverse, the related work [44] [6] [76] considers a point-to-point link and focuses on the TCP layer. The random access protocols [2] [80] [81] only work in the low load region and become unstable while operating in high load region. Connors [26] and Choi [34] focus on the MAC layer and their results are based on the end-to-end TCP. Mhatre [69] considers both the transport layer and the MAC layer problems. However the STP [44] adopted in his scheme still uses TCP congestion control algorithms which well known is not efficient and effective for the satellite connections. The web traffic used by Mhatre

is oversimplified.

We argue that the web browsing over multiple access reverse channel need to be addressed in a systematic manner from application layer, transport layer down to MAC layer. In the following chapters, chapter 3 will address the problems in the transport layer; chapter 4 will focus on the MAC protocol design for short messages such as HTTP requests in the reverse channel; finally in chapter 5, a closed queuing model is developed for capacity planning in both forward and reverse channels. The end-to-end web browsing performance is also evaluated with realistic application layer traffic model and different combinations of transport and MAC layer protocols.

Chapter 3

A Transport Layer Protocol in Satellite networks

Satellite networks are going to play an important role in the global information infrastructure. Satellites can be used to provide Internet services to fixed users and to mobile users as shown in figure 1.2. However, recent measurements show that the satellite link efficiency is only about 30% [3]. In order to improve the performance of Internet over satellite, a new transport layer protocol called receiver window backpressure protocol (RWBP) is proposed in this chapter. RWBP uses per-flow queuing, round robin scheduling and receiver window backpressure for congestion management. Analytical and simulation results show that RWBP can maintain high utilization of the satellite link and improve fairness among the competing connections.

Most of the Internet traffic is TCP traffic. TCP works well in terrestrial networks. However, TCP performance degrades dramatically in satellite networks. The measured efficiency 30% of the satellite forward channel is too low to satellite service providers. There are at least four aspects that cause the low efficiency:

First, the round trip time (RTT) is very large in satellite networks. The RTT in geo-synchronous orbit (GEO) satellite networks is about $500ms$. The time taken by TCP slow start to reach the satellite bandwidth ($SatBW$) is about $RTT * log_2(SatBW * RTT)$ when every TCP segment is acknowledged [77]. For a connection with large RTT, it spends a long time in slow start before reaching the available bandwidth. For short transfers, they could be finished in slow start, which obviously does not use the bandwidth efficiently. Some researchers propose to use a larger initial window [7] up to about $4K$ bytes rather than one maximum segment size (MSS) for slow start. So files less than $4K$ bytes can finish their transfers in one RTT rather than 2 or 3 RTTs. Another proposal [35] is to cancel the delayed acknowledgement mechanism in slow start so every packet is acknowledged and the sender can increase its congestion window (CWND) more quickly.

Second, Satellite channel is noisier than fiber channel. Bit error rates (BER) of the order of 10^{-6} are often observed [44]. Under the bad weather or when the terminals are mobile, bit error rates can be even higher. Because TCP treats all losses as congestion in the network, this kind of link layer corruption can cause TCP to drop its window to a small size and lead to low efficiency.

Third, congestion could happen in the reverse channel. The forward channel bandwidth from the satellite gateway to the ground terminals is much larger than the reverse channel bandwidth [8]. Most of the time, the users download data from the Internet and the reverse channel is used to transfer TCP ACKs. When the reverse channel traffic load is greater than its bandwidth, congestion could happen. The congestion in reverse channel may cause poor performance in the forward channel because TCP uses ACKs to clock out data. In the best case,

the ACKs are not lost, but queued, waiting for available bandwidth. This has a direct consequence on the retransmission timer and slows down the dynamics of TCP window. To alleviate this problem, ACK filtering [12] is proposed to drop the ACKs in the front of the IP queue by taking advantage of the cumulative acknowledgement strategy in TCP.

Lastly, the satellite link is shared by the TCP traffic and high priority traffic. Usually the satellite network operators provide both Internet services and other services such as multicasting video or audio. Because of the higher revenue brought by the multicasting traffic, it is given a higher priority. The leftover bandwidth is used by TCP traffic. To achieve high utilization of the satellite link, TCP traffic has to adapt its sending rate to fill in the leftover bandwidth. This problem has not been addressed in the literature so far. What is assumed implicitly in the literature is that the satellite link is used exclusively by TCP traffic.

In addition to the low efficiency, the satellite networks lack an effective fairness control mechanism. The fairness control policy of a real satellite network is total usage based [90]. If the total amount of data downloaded by a user exceeds a certain threshold for an extended period of time, the user's throughput could be throttled for several hours. Although this scheme can prevent abusive consumption of bandwidth by a small number of users, the time scale it operates on is too coarse and it cannot adapt to the traffic arrival pattern on small time scales.

Internet over satellite is more expensive than its terrestrial alternatives. In oder to provide competitive Internet services over satellite, a receiver window backpressure protocol (RWBP) is proposed in this chapter to improve the performance in term of both efficiency and fairness in satellite networks.

The rest of this chapter is organized as follows. Section 3.1 describes the

system model of the satellite networks. Section 3.2 presents the congestion control, error control, buffer allocation and management in RWBP. Section 3.3 gives the analytical and simulation results of RWBP. Section 3.4 describes the deployment considerations. Finally, section 3.5 concludes this chapter.

3.1 System Model

TCP connections in satellite networks need large windows to fully utilize the available bandwidth. However it takes much longer for satellite TCP connections than for terrestrial TCP connections to reach the target window size because of the long propagation delay and the slow start algorithm in TCP. Furthermore the multiplicative decrease strategy makes the hard gained TCP window very vulnerable to congestion. The misinterpretation of link layer corruption as congestion makes this situation even worse. In the best case, the packet loss does not cause timeout and TCP can stay in congestion avoidance phase rather than in slow start, the additive increase strategy makes the window to grow very slowly. Therefore TCP performance over satellite degrades dramatically.

Because the feedback information of the satellite networks is either delayed too long or too noisy, end-to-end schemes cannot solve these problems very effectively [44] [43]. An alternative to end-to-end schemes is to keep the large window of packets inside the network such as in the proxies located at the boundary of the satellite and terrestrial networks. Considering the interoperability issue, we adopts the connection splitting based scheme [90] [11] [20] [44] which is currently used in the industry, and we design a new protocol for reliable data transfer over the satellite link.

In the network as shown in figure 1.2, an end-to-end TCP connection is split

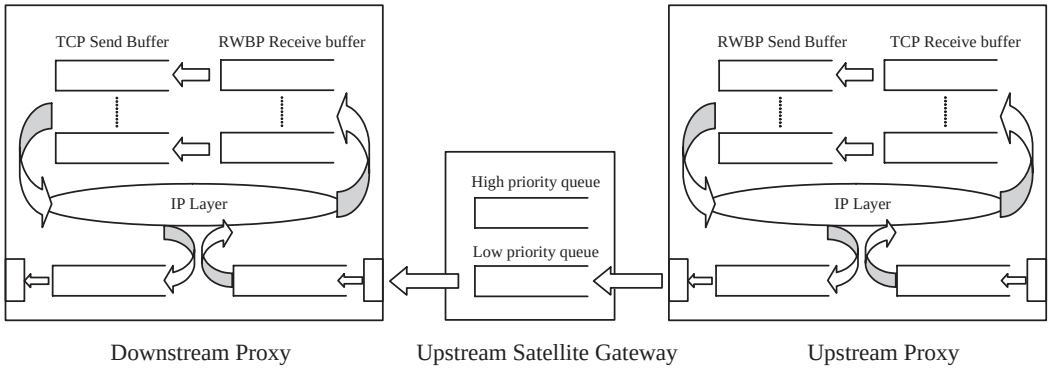


Figure 3.1: Queuing model for the satellite gateway and proxies. Flow control is done between downstream proxy and upstream proxy, and between satellite gateway and upstream proxy. It is also done between link layer and the IP layer, between the IP layer and transport layer, and inside transport layer.

into three connections at the proxies. The first one is set up between the server and the upstream proxy; the second is from upstream proxy to downstream proxy; and the third is from the downstream proxy to the client. Upstream proxy sends premature acknowledgements to the server and takes responsibility to relay all the acknowledged packets to the downstream proxy. Downstream proxy relays the packets to the client the same way as the upstream proxy relays the packets. Normal TCP is used for the server-proxy and proxy-client connections. Receiver window backpressure protocol (RWBP) is designed for the proxy-proxy connection to transfer data over the satellite link. RWBP has newly designed congestion control and error control algorithms, which can achieve high utilization of the satellite link and improve fairness among competing connections.

3.1.1 Queuing Model at the Satellite Gateway

The satellite gateway and the very small aperture terminals (VSAT) are connected to their local proxies (figure 1.2) through high-speed links whose bandwidth is much larger than the satellite link bandwidth. Therefore between the upstream and downstream proxies, the satellite link is the bottleneck link. The satellite link is used to transfer TCP traffic as well as multicasting video or audio traffic. At the satellite gateway, we assume that a high priority queue is used for multicasting traffic and a low priority queue is used for TCP traffic. These two queues are link layer queues at the terrestrial-satellite output interface (figure 3.1).

3.1.2 Queuing Model at the Proxy

In a traditional router, only those packets waiting for transmission are buffered in the IP output queue. However, the proxies have to buffer the packets waiting for transmission as well as those packets that have been transmitted but not acknowledged. A traditional router keeps all the packets in a FIFO queue while the proxies have a queue for each connection. From figure 3.1, we can see that the input queues at IP layer and link layer should be almost always empty if we assume that the proxy CPU processing rate is not the bottleneck. Therefore the possible queuing points at the proxies are transport layer receive/send buffer, IP output queue and link layer output queue.

3.2 Receiver Window Backpressure Protocol

Receiver window backpressure protocol (RWBP) is based on TCP with newly designed congestion control and error control algorithms. Although TCP congestion

control algorithms performs fine in terrestrial networks, it is not efficient and effective in satellite networks. Besides the inefficient congestion control algorithms, TCP windowing scheme ties the congestion control and error control together and errors can stop the window from sliding until they are recovered. The above observations motivate us to decouple error control from congestion control in TCP first and then design more efficient and effective congestion and error control schemes with the specific characteristics of satellite networks in mind. Our design goals of RWBP are to: 1) achieve high utilization of the satellite link; 3) improve fairness among competing connections; 3) reduce response time of short transfers such as web browsing; 4) reduce the reverse channel bandwidth requirement without the degradation of forward channel performance.

3.2.1 Congestion Control

TCP uses slow start to probe the bandwidth at the beginning of a connection and uses additive increase and multiplicative decrease (AIMD) congestion avoidance to converge to fairness in a distributed manner. RWBP is based on TCP; however RWBP cancels all the congestion control algorithms in TCP and uses per-flow queuing, round robin scheduling [22] [58] [57] [56] and receiver window backpressure for congestion control (figure 3.1).

Flow control is done between the downstream proxy and the upstream proxy at the transport layer by using the receiver window (figure 3.1). For each RWBP connection, the downstream proxy advertises a receiver window based on the available buffer space for that connection just as in TCP. RWBP does not use Window scaling to advertise large windows to upstream proxy because large window scale factor can produce inaccurate values. In RWBP, the 16-bit receiver window field

is still used but its unit is maximum segment size rather than byte. Similarly flow control is also done between the satellite gateway and the upstream proxy at the link layer (figure 3.1). The low priority queue at the satellite gateway advertises a receiver window to the upstream proxy so that the low priority queue will not overflow.

In addition, flow control is done between the transport layer and the IP layer, and between the IP and the link layer. At the upstream proxy, a round-robin scheduler can send a packet for a RWBP connection only if the available advertised receiver window is greater than zero and there is at least one packet buffer space available at the IP output queue. When there is no packets can be sent or the available advertised receiver window is zero, the scheduler goes on to serve the next connection. When the IP layer output queue sends packets to the link layer, it has to make sure that the link layer queue is not going to be overflowed. This allows the link layer congestion backpressure to propagate to IP layer and then to transport layer. Inside the transport layer, when packets are moved from upstream connection receive buffer to the downstream send buffer, a blocking write is performed so that the send buffer will not overflow. This way the congestion is back pressured to the receive buffer of the upstream connection and a smaller receive window is going to be sent to the source. Finally the congestion is back pressured to the source. When the traffic load decreases, the buffers begin to be emptied faster and larger advertised receiver windows will be sent to the sources so the sources can speed up. If some connections are bottlenecked upstream or are idle because the application layers do not have data to send, the scheduler can send packets from other connections and high satellite link efficiency can be achieved. The round-robin scheduler does not take into account the packet sizes. Connec-

tions with larger packet sizes can get more bandwidth than those with smaller packet sizes. This problem can be solved by a more sophisticated scheduler and is left as future work.

The above flow control scheme can guarantee that there is no buffer overflow in the downstream proxy queues, in the upstream proxy queues or in the low priority queue at the satellite gateway. Therefore if a packet loss is detected at downstream proxy, it must be due to satellite link corruption rather than due to buffer overflow. Therefore RWBP decouples error control from congestion control.

3.2.2 Error Control

TCP depends on duplicate acknowledgements and timer for error control. Because out of order packet arrivals are possible in the wide area networks, the fast retransmit algorithm is triggered after three rather than one or two duplicate acknowledgements are received. The high bit error rate of the satellite link can cause multiple packet losses in a single window and may lead to timeout. In addition the loss probability over the satellite link is determined by the bit error rate and packet size, so the retransmission packets can be corrupted as probable as original packets when the error rate is high [83]. When the retransmitted packets are lost, timer could be the only means for error recovery in TCP. However, the timeout value is usually set much larger than the round trip delay to make sure the original packet does leave the networks to avoid false retransmission. The conservative loss detection and recovery schemes in TCP are not effective in satellite networks.

In RWBP, we explore the specific characteristics of the satellite networks. Firstly, because RWBP congestion control can eliminate packet losses due to buffer overflow, any loss must be caused by the link layer corruption. So the error control

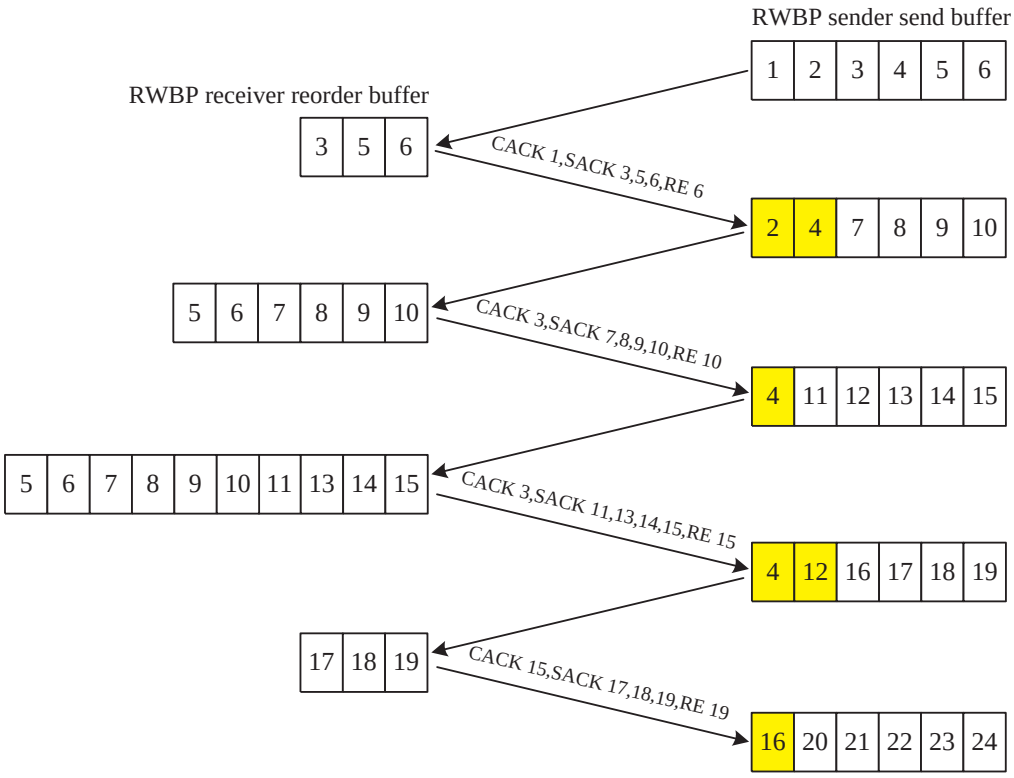


Figure 3.2: Error control in RWBP. CACK : Cumulatively acknowledged; SACK : Selectively acknowledged; RE : Right edge

scheme can operate independently with the congestion control scheme. Secondly, the satellite link is a FIFO channel and out of order packet arrivals are impossible. RWBP error control algorithms explore the in order packet delivery characteristic for error detection and use selective acknowledgement (SACK) for error recovery.

All data packets including retransmission packets of a RWBP connection are sorted in their transmission order. RWBP sender keeps track of the right edge packet in sequence space of all acknowledged packets, i.e. cumulatively or selectively acknowledged packets. Whenever an acknowledgement packet is received, RWBP sender compares in sequence space the right edge packet acknowledged in the current ACK with that in the previous ACK. If the sequence number does

not advance, RWBP error control algorithm does nothing. Whenever the sequence number advances, RWBP error recovery scheme is triggered. The first match of the current right edge packet in the sorted list must have arrived at the RWBP receiver. If a packet before the right edge packet in the sorted list is neither cumulatively acknowledged nor selectively acknowledged, RWBP assumes that the packet is lost and retransmits it. From the following example and the simulation results in section 3.3, we will see RWBP can recover not only the first time losses but also the retransmission losses very effectively. Timer is still used as the cast resort for loss recovery. After timer expires, two copies of the lost packet are sent to increase success probability.

Figure 3.2 gives an example of RWBP error control mechanism. On the left of figure 3.2, in sequence packets are not drawn and only out of order packets are drawn. Initially the right edge packet is set to packet 0. Packet 1, 2, ..., 6 are sent to the RWBP receiver. Packet 2 and packet 4 are corrupted. The receiver acknowledges packet 1, 3, 5 and 6. The sender compares the right edge packet 6 in the ACK to the initial right edge packet 0. The right edge packet advances in sequence space. And the sender checks the sorted list and finds out packet 1, 2, 3, 4 and 5 are sent before packet 6. Only packet 1, 3 and 5 are acknowledged, packet 2 and 4 should be lost. Packet 2 and packet 4 are retransmitted before new packet 7, 8, 9 and 10. Packet 2 arrives at the receiver successfully; however packet 4 is corrupted again. The receiver cumulatively acknowledges up to packet 3 and selectively acknowledges packet 7 to packet 10. The right edge packet advances in the sequence space from packet 6 to packet 10. The sender checks the sorted list and finds out packet 4 is transmitted before packet 10. However packet 4 has not been acknowledged, therefore packet 4 should be lost again. Packet 4 is

retransmitted again. The above procedure keeps on going until the sender finishes its transfer. It is not difficult to see from this example that RWBP can not only recover effectively the first time losses such as packet 2 and packet 12, but also the higher order packet losses such as packet 4, which has been corrupted three times.

When a packet is received correctly by the RWBP receiver but all the acknowledgments for it are lost, RWBP could retransmit this packet unnecessarily. In RWBP, one acknowledgement can carry up to four SACK blocks. As long as the acknowledgements are not sent very infrequently, this event should be rare.

3.2.3 Buffer Allocation and Management

The buffer sizes allocated to each connection at the upstream and downstream proxies have a direct impact on the end-to-end throughput. Firstly, assume that there is only one end-to-end transfer in the system. The satellite link is error free and its bandwidth is $SatBW$. At the upstream proxy, the size of the TCP receive buffer is $RecvBuf$ and the size of the RWBP send buffer is $SndBuf$. The round trip time of the satellite RWBP connection is $SatRTT$ and the round trip time of the terrestrial TCP connection is $TerrRTT$. Then the maximum achievable throughput of the end-to-end transfer is $MIN(SatBW,\ SndBuf/SatRTT,\ RecvBuf/TerrRTT)$. From this formula, we can see that $SndBuf$ or $RecvBuf$ can become the bottleneck of the end-to-end performance if it is less than the bandwidth delay product of its corresponding connection. However if the buffer size is greater than the bandwidth delay product, the satellite link becomes the bottleneck and packets could be backlogged at the proxy. The same analysis applies to the downstream proxy.

When there are multiple connections in the system, the bandwidth available

to each connection depends on the number of connections and their activities. One possible buffer allocation scheme is the adaptive buffer sharing [22] which dynamically allocates a buffer pool to all the connections based on their bandwidth usage. While this scheme can dramatically decrease the buffer requirement, it is complex to be implemented. In RWBP, each connection is assigned a static peak rate and the buffer size is set to the peak rate delay product (PRDP).

When the satellite link is error free, the buffer sizes allocated above are enough to achieve the target peak rate. However when the satellite link is error prone, changes need to be made to the buffer sizes at both the downstream and the upstream proxies.

When a packet is corrupted, the downstream proxy has to buffer the out of order packets because RWBP receiver only forwards in sequence packets. For example, in figure 3.2 the in sequence packet 1 is forwarded while the out of order packets 3, 5, and 6 are kept in the receive buffer. In order to keep the advertised receiver window open so that the RWBP sender can send new packets during the error recovery, the downstream proxy needs a buffer size larger than the peak rate delay product to achieve the peak rate. If the error rate of the satellite link is low and corrupted packets can be recovered in one RTT, receive buffer size about two times of the peak rate delay product should be provided. If the error rate is relatively high, retransmissions packets can be corrupted. Our simulation results in section 3.3 show that receive buffer size should be set to about three to four times of the peak rate delay product to maintain high satellite link utilization.

For the upstream proxy, the buffer management in RWBP is different from that in TCP SACK [66]. In TCP SACK, only packets cumulatively acknowledged are released from the send buffer. Packets selectively acknowledged are still kept

in the send buffer because the TCP receiver may renege and discard the SACKed packets when it runs out of receive buffer. For example, in figure 3.2 cumulatively acknowledged packet 1 is released from the send buffer, however selectively acknowledged packet 3, 5 and 6 are still kept in the TCP send buffer. The buffer occupied by these SACKed packets can cause the upstream proxy to advertise a smaller window to the source. This will slow down or even stall the source. After the error is recovered, the cumulative acknowledgement may clear a large number of packets from the proxy send buffer. The upstream proxy could run out of packets to send and it has to wait for new packets to arrive from the source. Therefore the terrestrial TCP connection could cause starvation of the upstream proxy queue. In RWBP, the receiver never reneges and sender does not clear the SACK state information after timeout. So the SACKed packets can be released from the send buffer. Thus only those packets actually corrupted over the satellite link are still kept in the buffer. For example, in figure 3.2, successfully received packet 3, 5 and 6 are released from the buffer and only the lost packet 2 and 4 are still buffered. The buffer management mechanism in RWBP solves the stall and starvation problems in TCP SACK mentioned above, which contributes to the end-to-end performance improvement.

3.3 Performance Evaluation

In this section, we first model the performance of RWBP for both bulk and short transfers. Next our model is verified by packet level simulations with OPNET. The metrics we are interested in are reverse channel bandwidth requirement, end-to-end throughput and fairness for bulk transfers, response time for short transfers and satellite link utilization when there is high priority traffic competing with TCP

traffic.

3.3.1 Performance Modeling of RWBP

Efficiency and fairness of bulk file transfers

In the following, we assume that there are N end-to-end persistent FTP transfers in the system and there are no link corruption over the satellite link. The satellite link bandwidth is $SatBW$. We also assume that the only possible bottleneck in the satellite network is the upstream proxy. Therefore the downstream proxy has no effect on the end-to-end performance. The RTT in the satellite networks is $SatRTT$ which is the same for all transfers and the RTT for transfer i in the terrestrial networks is $TessRTT_i$. The loss probability of transfer i at the satellite gateway is p_i and the loss probability of transfer i in the terrestrial networks in p'_i. We assume the two loss events are independent of each other. From TCP modeling [75], we can get the throughput of an end-to-end TCP transfer

$$E2E\_TCP\_Throughput(i) = \frac{C * MSS}{(SatRTT + TerrRTT_i) * \sqrt{p_i + p'_i - p_i p'_i}} \quad (3.1)$$

Where C is a constant and MSS is the maximum segment size. When the congestion loss rate is relatively large which are further magnified by the large round trip time, the end-to-end TCP throughput could become much less than its share of the satellite bandwidth i.e.

$$E2E\_TCP\_Throughput(i) \ll \frac{SatBW}{N} \quad (3.2)$$

Therefore

$$\sum_{i=0}^{N} E2E\_TCP\_Throughput(i) \ll SatBW \quad (3.3)$$

Formula 3.3 shows that the aggregate throughput of the N end-to-end transfers is much less than the satellite link bandwidth therefore the satellite link is not used efficiently.

Because $TerrRTT_i$, p_i and p'_i are usually different from each other for different transfers, in general the throughput of transfer i is not equal to the throughput of transfer j i.e.

$$E2E\_TCP\_Throughput(i) \neq E2E\_TCP\_Throughput(j) \tag{3.4}$$

The above formula shows that the networks resources are not shared fairly among all the transfers.

In RWBP, there is no congestion loss at the satellite gateway i.e. $p_i = 0$, the throughput of the terrestrial connection i is

$$RWBP\_Terr\_Throughput(i) = \frac{C * MSS}{TerrRTT_i * \sqrt{p'_i}} \tag{3.5}$$

Because $TerrRTT_i$ and p'_i are both small, the terrestrial connection throughput can be larger than its satellite bandwidth share. In equilibrium state, be have

$$RWBP\_Terr\_Throughput(i) = \frac{SatBW}{N} \tag{3.6}$$

The aggregate throughput of the N transfers in RWBP is

$$\sum_{i=0}^{N} RWBP\_Throughput(i) = SatBW \tag{3.7}$$

This shows that the satellite link is fully utilized. Besides each transfer gets the same throughput of $SatBW/N$, therefore the total bandwidth is fairly allocated to all transfers. From the comparison above, we can see that RWBP can improve both the efficiency and fairness performance for bulk transfers. In the following we will model the response time of RWBP for short transfers.

Response time of short transfers

In order to simplify the analysis, we assume the Internet server is located near the upstream proxy and the $TerrRTT$ is negligible and we consider only one transfer. We also assume transaction based TCP is used in which the short transfer request is sent together with the TCP SYN packet. Assume the packet size is 512 bytes and the file size is 11 packets. We also assume there is no congestion and link layer loss in this section.

In end-to-end TCP with delayed acknowledgement strategy, four $SatRTT$[1] plus additional transmission time $Trans_{E2E}$ are needed for the whole file transfer.

$$E2E\_resp\_time = 4 * SatRTT + Trans_{E2E} \tag{3.8}$$

while in RWBP, there is no slow start and the packets are push to the terminals once the request is received. The response time in RWBP is

$$RWBP\_resp\_time = SatRTT + \frac{FileSize}{AverageThroughput} \tag{3.9}$$

If we assume the average throughput at the upstream proxy in RWBP is 600 kbps, the $RWBP\_resp\_time$ is 575 ms. Therefore the response time improvement of RWBP over end-to-end TCP is $E2E\_resp\_time/RWBP\_resp\_time$ which is about 3.5 times better.

The performance modeling in the above two sections shows that RWBP outperforms end-to-end TCP for both bulk transfers and short transfers. In the following, we will give more detailed performance evaluation of RWBP with packet level simulations which will verify the results we get in the modeling.

[1]The number of packets transmitted in the four RTTs is 1, 2, 3, 5.

3.3.2 Proxy Buffer Size and Reverse Channel Bandwidth Dimensioning

In order to achieve high throughput in the forward channel, we need to find out the proxy buffer size and reverse channel bandwidth requirements. If the requirements are not satisfied, they could become the system bottlenecks.

In figure 1.2, a single transfer in the system is set up between a client and an Internet server. The satellite link bandwidth is 600kbps. Server-proxy and proxy-client link bandwidth is 2Mbps. The RTT of the proxy-proxy connection is 500ms, the RTT of the server-proxy connection is 80ms and the RTT of the proxy-client is 10^{-4}ms. The packet size is 512 bytes and the file size is 3M bytes. The peak rate is set to the satellite link bandwidth. To get the downstream proxy receive buffer size requirement, it is set to infinity and all the other proxy buffer sizes are set to the peak rate delay product.

In RWBP, an acknowledgement is sent when every N data packets are received. By changing N, we can change the acknowledgement frequency. Figure 3.3 shows that when N increases exponentially i.e. the ACK frequency decreases exponentially, the reverse channel bandwidth usage decreases exponentially. However the forward channel throughput is very insensitive to the reverse channel usage (figure 3.4). Only when N increases up to 16, the forward channel throughput begins to decrease. This happens because of the following reason. Although ACKs are not used to clock out data packets in RWBP, they are still used to clear upstream proxy buffers. Less frequent ACKs can cause the buffers to be filled up and to slow down the terrestrial connections. When N equals eight, the forward channel throughput is very close to that achieved when N equals one. Therefore we set N to eight in RWBP. In TCP, one ACK is sent every two data packets are received.

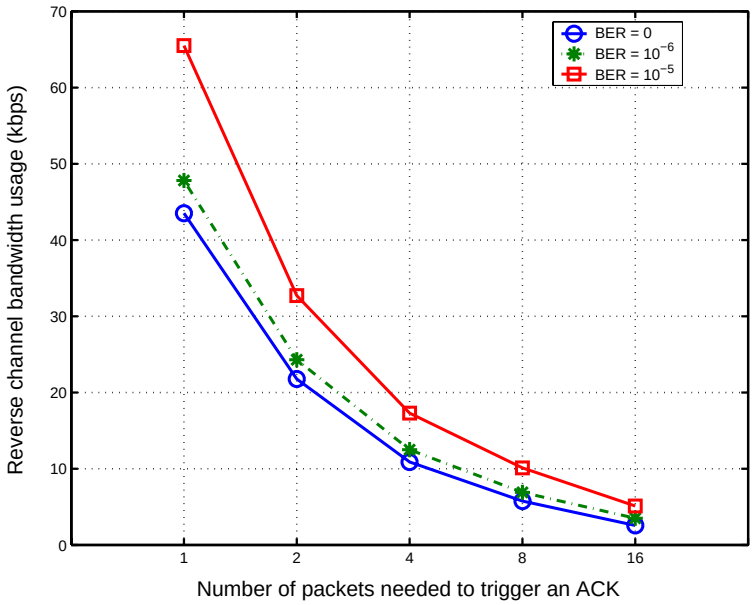


Figure 3.3: Reverse channel bandwidth usages for different acknowledgement frequency in RWBP

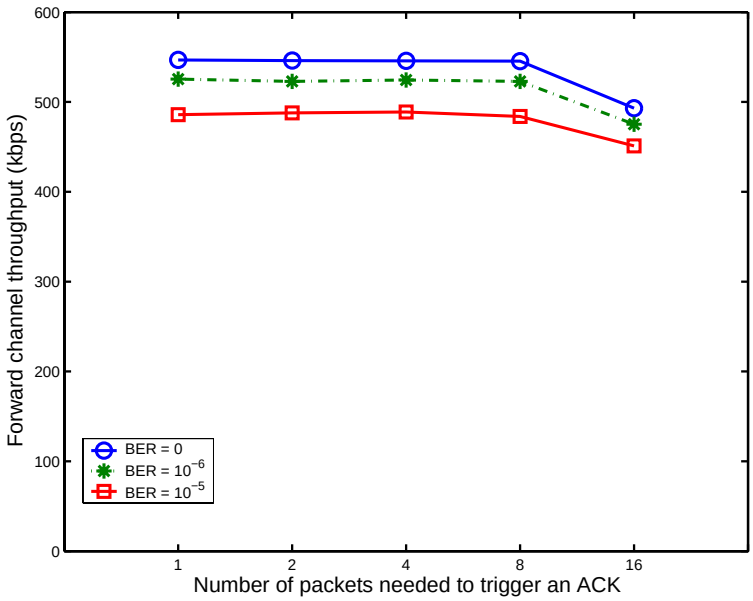


Figure 3.4: Forward channel throughput for different acknowledgement frequency in RWBP

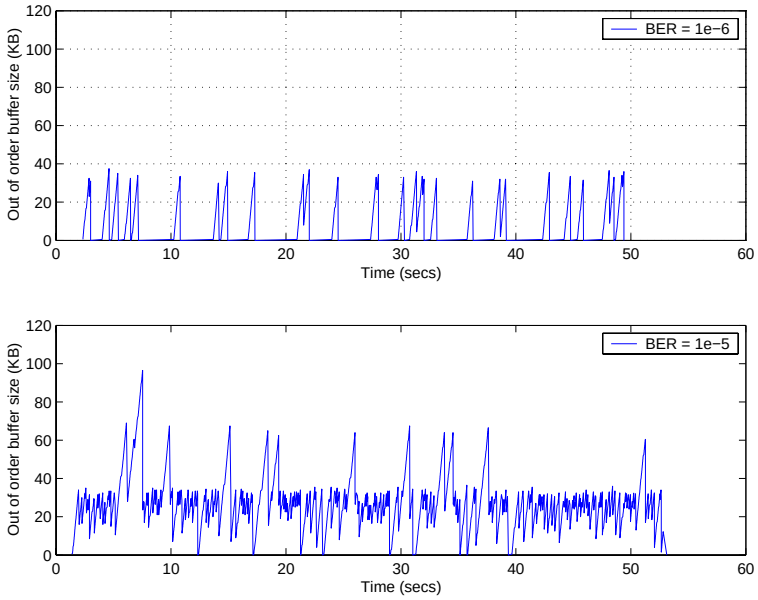


Figure 3.5: Reorder buffer size at the downstream proxy for BER = 10^{-6} and BER = 10^{-5}

So RWBP can reduce the reverse channel bandwidth requirement to one quarter of that in TCP without sacrificing the forward channel throughput.

Figure 3.5 shows the reorder buffer sizes at the downstream proxy for bit error rate equals 10^{-6} and 10^{-5}. For both cases, one acknowledgement is sent when every eight data packets are received. For BER equals 10^{-6}, occasionally there is about one peak rate delay product (PRDP) of packets in the reorder buffer (upper plot in figure 3.5). This means that the errors can be recovered in one RTT. However when the error rate is increased to 10^{-5}, retransmissions can be lost too. The lower plot in figure 3.5 shows that retransmissions could be lost twice because sometimes there are about three PRDP of packets in the reorder buffer. Therefore in order to achieve high forward channel throughput, downstream proxy receive buffer size larger than PRDP is needed so that the advertised window can remain open and the upstream proxy can continue to send new packets during error recovery. For

low bit error rate, buffer size about two times of the PRDP is needed. While for high bit error rate, buffer size about four times of PRDP is needed (figure 3.5).

3.3.3 Throughput and Fairness for Bulk Transfers

In figure 1.2, fifteen clients download large files from fifteen Internet servers. The satellite bandwidth is 9Mbps. The link bandwidth from each server to the upstream proxy is 2Mbps. The link bandwidth from downstream proxy to each client is also 2Mbps. The *RTT*s for all the fifteen proxy-proxy connections are 500ms. The RTTs of the fifteen proxy-client connections are all set to 10^{-4}ms. The *RTT* of the server-proxy connection corresponding to end-to-end transfer i is (20*i-16)ms. Therefore the end-to-end round trip time for transfer i is 500.1 +(20*i-16)ms, i.e. in the range [504.1ms, 784.1ms]. The peak rate is set to 1.2Mbps. The downstream proxy receive buffer size is set to two times peak rate delay product(PRDP) and all the other proxy buffer sizes are set to one PRDP. The satellite gateway buffer size is set to the satellite bandwidth delay product.

End-to-end throughput for bulk transfers

In figure 3.6, we compare the end-to-end aggregate throughput of RWBP and TCP SACK for different bit error rates when they are used for the proxy-proxy connections. All the terrestrial connections use TCP Reno. The simulation time is 1000 seconds.

When the bit error rate is very low, both schemes can achieve very high throughput. For TCP SACK when the bit error rate increases up to 10^{-6}, the link layer corruption causes the upstream proxy TCP to drop its congestion window and leads to degraded performance. When the loss rate is increased further to 10^{-5},

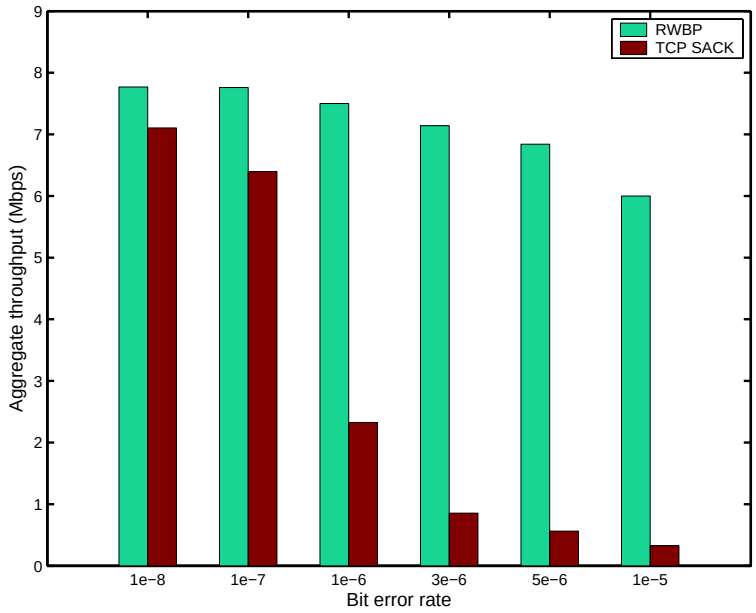


Figure 3.6: Aggregate throughput for different bit error rates in RWBP and TCP

the retransmitted packets can get lost again and TCP SACK may have to wait for the timeout to recover the losses. After timeout, the congestion window is set to one and TCP enters slow start. Therefore the satellite link utilization is very low for high loss rate when TCP is used. In RWBP, the congestion control is decoupled from the error control. Because the upstream proxy can schedule new packets to be sent during error recovery and RWBP error control can recover effectively first time as well as higher order losses, RWBP can achieve higher throughput for both low and high bit error rates (figure 3.6).

Fairness for bulk transfers

A 6M bytes file is downloaded from Internet server i to client i. Figure 3.7 plots the received packet sequence number growth at the clients for the fifteen transfers. For BER equals 10^{-6}, the upper plot in figure 3.7 shows that the sequence numbers

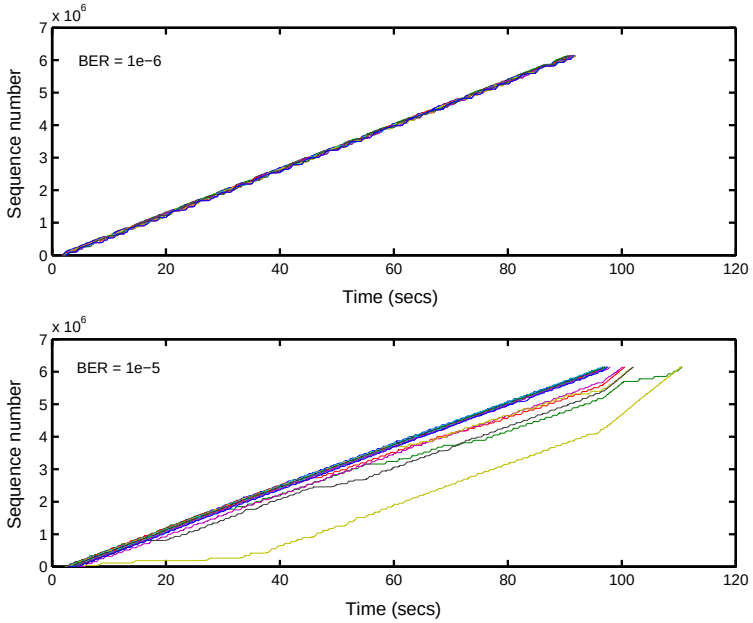


Figure 3.7: Received sequence number progress at the fifteen clients for BER = 10^{-6} and BER = 10^{-5} in RWBP

of the fifteen transfers grow almost at the same rate because they are overlapping with each other. Therefore data packets arrive at the clients almost at the same rate and each transfer gets a fair share of the satellite link bandwidth. When BER increases to 10^{-5}, the sequence numbers still grow at close rates. For the lowest curve in figure 3.7, at the beginning of this transfer, the sequence number grows at a much lower rate than those of other transfers. The reason is that its errors cannot be recovered by RWBP error control algorithm and the upstream proxy has to wait for the timer to expire. After about 35 seconds, this transfer recovers and its packets arriving rate becomes close to those of other transfers.

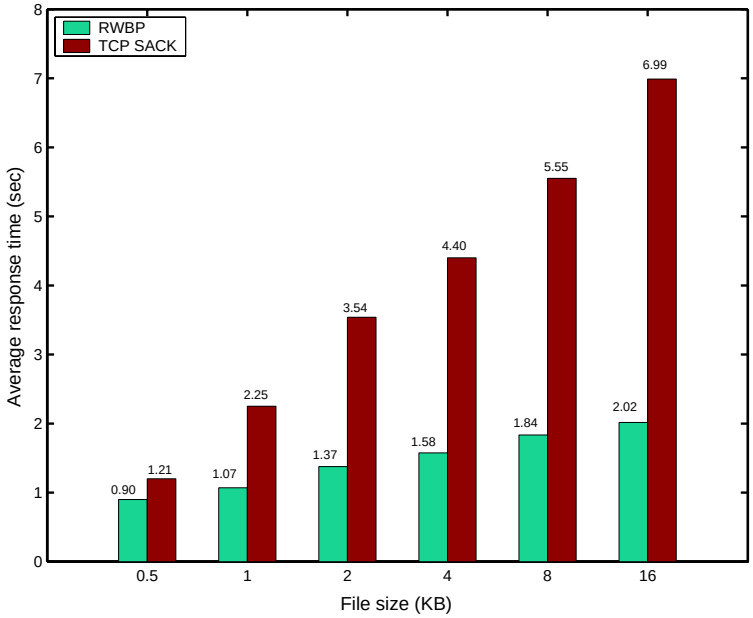


Figure 3.8: Response time for short transfers in RWBP and TCP

3.3.4 Response Time for Short Transfers

In addition to bulk file transfers, another popular application is web browsing, which is characterized by the clients send small requests and the servers reply with small files. The same network configuration is used as in section 3.3.3 and the BER is 10^{-6}. All the transfers between servers and clients are still bulk transfers except that the bulk transfer between server five and client five is replaced by short file transfers. Client five randomly requests small files of fixed size from Internet server five. Figure 3.8 shows the average response time for different file sizes. The average response time is calculated over 1000 samples. RWBP performs better than TCP SACK for the following reasons. Firstly, RWBP does not need to go through the slow start phase and packets can be sent as long as the link is available. Secondly, because RWBP provides per-flow queuing, packets of short transfers do not need to wait after packets of bulk transfers in the FIFO queue

at satellite gateway. Therefore, RWBP isolates short transfers from bulk transfers and decreases the queuing delay of short transfers.

3.3.5 Satellite Link Utilization

In the above evaluations, the satellite link is used exclusively by TCP traffic. However in practice, the satellite bandwidth is shared between TCP traffic and high priority multicasting UDP traffic. In this section, we will evaluate how RWBP performs under situation where there is competing high priority traffic. It is also noticed that in the above the satellite link is assumed to be the only bottleneck in the system. In the following we will evaluate a more realistic scenario where downstream and upstream bottlenecks exist.

Satellite Link Utilization with Competing High Priority Traffic

We use the same network set up as in section 3.3.3. However only the transfer between server five and client five is activated. The satellite link bandwidth is set to 600kbps. The peak rate is set to 600kbps. The downstream proxy receive buffer size is set to two times PRDP and all the other proxy buffer sizes are set to one PRDP. The satellite gateway buffer size for low priority TCP traffic is 75 packets, which is about the satellite bandwidth delay product and the buffer size for high priority UDP traffic is 50 packets. The bit error rate of the satellite link is set to zero. We compare the satellite link utilization when RWBP and TCP SACK are used for the proxy-proxy connection.

TCP transfer starts at the 150th second and a file of 36M bytes is sent from server five to client five. UDP traffic begins at the 240th second and ends at the 960th second. Firstly when only TCP traffic is active, the upper plot in

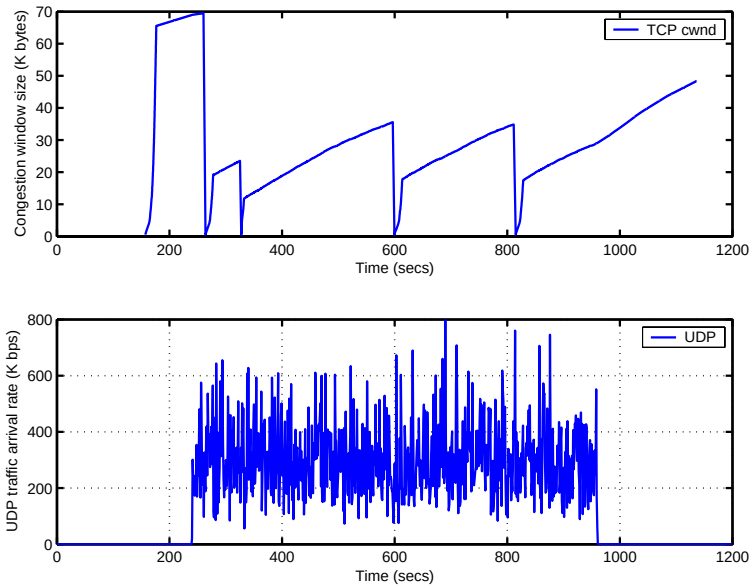


Figure 3.9: TCP congestion window size at upstream proxy and the arrival rate of UDP traffic at the satellite gateway

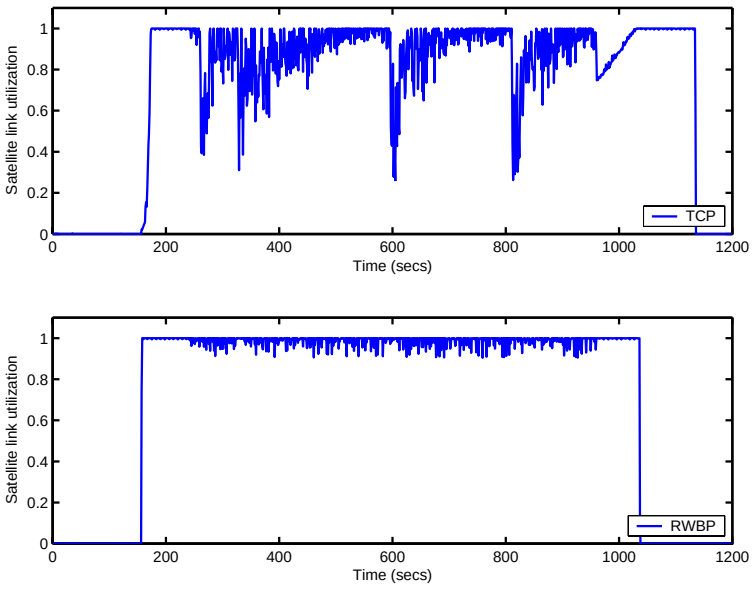


Figure 3.10: Satellite link utilization for TCP and RWBP with high priority UDP traffic

figure 3.9 shows that TCP can increase its congestion window large enough so that the satellite link bandwidth is fully utilized (upper plot in figure 3.10). However after a high priority UDP flow with average arrival rate of 300kbps (lower plot in figure 3.9) begins, its dynamically changed traffic demand causes low priority TCP periodically timeouts. After timeout, TCP goes into the slow start phase. Because of the long propagation delay of the satellite link, it takes a long time for TCP to increase its window large enough to fully utilize the satellite bandwidth. The upper plot in figure 3.10 shows that the satellite link utilization is low when there is competing UDP traffic with TCP traffic. However, RWBP can adapt to the high priority traffic load very well and the satellite link utilization is kept very high as shown in the lower plot of figure 3.10. Because link layer flow control is used between the satellite gateway and the upstream proxy, the congestion at the satellite gateway caused by the increase demand of high priority traffic is back pressured to the upstream proxy by advertising a smaller window. Because of this, the increasing rate of the high priority UDP traffic will not cause RWBP packets dropped at the satellite gateway. When the traffic demand of the UDP traffic decreases, a larger window is advertised to the upstream proxy so that RWBP can speed up to fill in the left bandwidth.

Satellite Link Utilization with Upstream and Downstream Bottlenecks

We use the same network set up as in section 3.3.3. Persistent traffic sources are used and the bit error rate of the satellite link is set to zero. Figure 3.11 shows the throughput for the fifteen transfers. Throughput is averaged over 4 seconds which are about five times of the largest end-to-end delay (784.1ms). At the beginning of this experiment, all the terrestrial link bandwidth is 2Mbps. After about the

fifteenth second, the throughput of all the transfers converge to about 550Kbps. At the seventieth second, the proxy-destination link bandwidth of transfer 1 to transfer 5 is set to 450Kbps. The first plot in figure 3.11 shows that transfer 1 to 5 are bottlenecked by these downstream links. However the throughput of the left ten transfers ramp up quickly so that the satellite link is still fully utilized (lower plot on figure 3.12). After the 130th second we set the proxy-destination link bandwidth back to 2Mbps, it is shown in figure 3.11 that all the transfers converge to the fair share of the satellite bandwidth again. From the 190th to the 250th second, we set the source-proxy link bandwidth of transfer 11 to transfer 15 to 450Kbps. Figure 3.11 shows that RWBP with upstream bottlenecks behaves almost the same as with downstream bottlenecks. From the 310th to the 370th second, we set the above two sets of links bandwidth to 450Kbps. In this case, the bandwidth received by transfer 6 to transfer 10 increases to about 830Kbps. The upper plot in figure 3.12 shows the throughput of transfer 1, 6 and 10. This experiment shows that RWBP can adapt to the upstream and downstream bottlenecks and maintain high utilization of the satellite link as shown in the lower plot in figure 3.12.

3.4 Deployment Considerations

From the performance evaluation section, we can see that RWBP outperforms TCP in term of both efficiency and fairness which is realized by the proxies on the boundary between the satellite and terrestrial networks. However the end-to-end semantics in TCP is changed and the processing overhead is more that required in a normal router. Another big concern is how RWBP interacts with security mechanisms on the Internet. In this section, we will discuss these deployment issues.

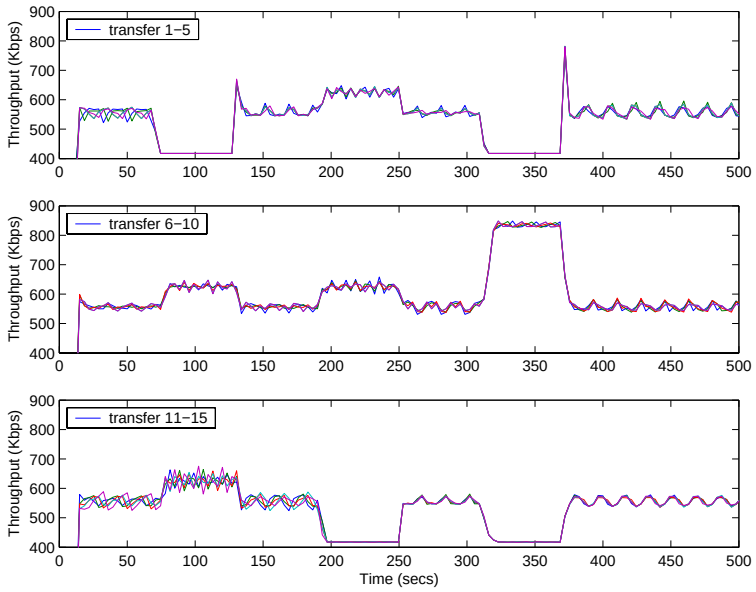


Figure 3.11: Throughput for different transfers with upstream and downstream bottlenecks

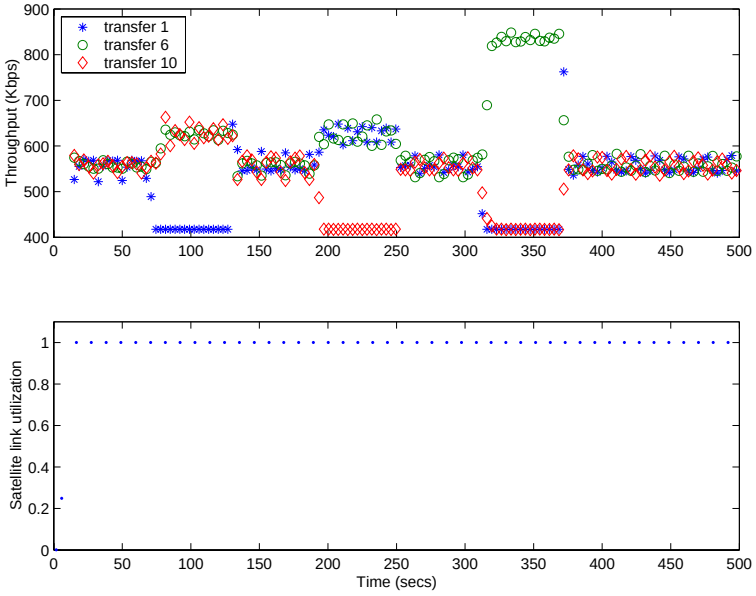


Figure 3.12: Throughput for transfer 1,6 and 10 and satellite link utilization

3.4.1 End-to-end Semantics and Reliability

RWBP uses the concept of connection splitting which violates the end-to-end semantics of TCP. It is possible that the sender receives an acknowledgement of a data packet while the data packet has not reached the final destination rather is buffered at the upstream proxy. The authors of I-TCP [11] argue that many applications such as FTP and HTTP use application layer acknowledgements in addition to end-to-end TCP acknowledgements. Using connection splitting for these applications does not comprise end-to-end reliability.

Although we implement RWBP in the transport layer, it is possible to implement it in the link layer so that a reliable link layer is used to transfer TCP traffic. While other traffic such as UDP traffic which does not require reliability can use another link layer protocols. The backpressure mechanism in RWBP should still work for this kind of implementation.

3.4.2 Security Issues

Because satellite proxy needs to access the TCP header for connection splitting, it will not work if IPSEC is used. One possible solution is the layered IPSEC technique proposed by Zhang [94]. TCP header of a packet is encrypted with one key, and the data of the packet is encrypted with another key. The satellite proxy only has the key to decrypt the TCP header. Therefore the proxy can still get the TCP header information and prematurely acknowledge the received data packets while at the same time the security is not sacrificed .

RWBP can coexist with any upper layer security protocol such as PGP in the application layer and SSL in the transport layer. For the same reason, RWBP can work with any link layer security protocol.

3.4.3 CPU Processing Overhead

Because the satellite link is still a scarce resource, loss-less compression can be used to improve the efficient utilization of the satellite link. The encryption, compression and checksum computation for connection splitting are all CPU intensive operations. It has been shown that the processing time without compression and encryption is small and a moderate machine can adequately support numerous split connections with little performance degradation [20].

If the satellite link bandwidth is large, several proxies can be used for one satellite gateway. The satellite gateway keeps a queue for each proxy and receiver window flow control is enforced between the satellite gateway and each proxy. Future work will address this scalability problem by taking into account all the processing overhead.

3.5 Summary

In order to improve the efficiency of satellite link and to provide effective fairness control, a new protocol called RWBP is proposed in this chapter. RWBP is designed for the satellite connections by taking advantage of the specific characteristics of the satellite networks. It uses per-flow queuing, round robin scheduling and receiver window backpressure for congestion management. The congestion management algorithms in RWBP can eliminate buffer overflows inside the satellite networks even when there is high priority traffic competing with TCP traffic. Therefore any loss inside the satellite networks must be caused by link layer corruption rather than by buffer overflows. So the error control in RWBP can operate independently with its congestion management. The newly designed error control

scheme in RWBP can effectively recover not only first time losses but also higher order losses. Our analytical and extensive simulation results show that RWBP can improve the performance of both bulk and short transfers over satellite networks.

We conclude that RWBP is a promising transport layer protocol for satellite networks especially networks with large propagation delay, high bit error rate, high priority traffic competition and reverse channel congestion. In this chapter, we assume there is a dedicated reverse channel for each terminal or gateway. This is true for the point-to-point topology and even true for the star topology if a telephone line is used for the reverse channel. However, in the star topology if all the terminals share a common reverse channel, the bandwidth assigned to each terminal is not necessarily fixed. In the next chapter, we will investigate the multiple access problem in the reverse channel for short messages.

Chapter 4

A Multichannel Random Access Protocol for Multislot Short Messages with Delay Constraints

A multichannel random access protocol called FMCSA for multislot short messages is proposed in this chapter to be used in access networks with large propagation delay. This protocol combines random access with the use of packet level forward error correction coding for new messages and scheduled retransmissions for partially received messages. Analytical and simulation results show that FMCSA can achieve a higher throughput and lower delay than slotted Aloha. When the system is operating at the low load region, the short messages can be delivered in their first attempts with very high probability. With the load increasing, more messages will be received partially in their first attempts and the scheduled retransmission scheme will guarantee the partially received messages to be recovered in their second attempts. Therefore the delay performance of FMCSA is much more robust to the load fluctuation than slot Aloha.

4.1 Traffic and Channel Models

We consider short message transfer in the reverse channel over a star satellite network. Short messages such as HTTP requests have a typical size about 400 bytes [65] [23] [10] [86]. At the MAC layer, a short message is fragmented into multiple smaller MAC packets. For example, assume each MAC layer slot carries a MAC header of 5 bytes and a MAC payload of 48 bytes, therefore a short message of 432 bytes will be segmented into 9 MAC packets. Only after all the MAC packets of a message are received will the hub reassemble the message and forward it to the upper layer.

All the terminals and the hub are synchronized and MAC packets are allowed to be sent only at the beginning of a time slot. We assume there are no other errors except the collisions caused by more than one MAC packets sent in the same slot. Whenever there is a collision, all the MAC packets involved are destroyed. If there is only one MAC packet sent in a slot, the packet will be received correctly by the hub and a positive acknowledgement will be sent back to the terminal through the forward channel. A selective reject retransmission strategy is used for collision recovery. Therefore if an acknowledgement is not received for a collided MAC packet after the timeout, only the collided MAC packet will be retransmitted until it is received correctly by the hub. All new messages generated by the upper layer are buffered at the MAC layer queue even when a terminal is backlogged and is attempting to retransmit a previous MAC packet [18].

We are interested in a wideband reverse channel which serves a large number of terminals. With the increase of the bandwidth, the transmission time per bit is decreased therefore the peak power of the transmitter has to be increased to keep the energy per bit constant [2]. However safety and cost constraints set a upper

limit on the peak power of a terminal transmitter. A straightforward way to extend a narrowband channel to wideband can be done by dividing the wideband channel into a number of narrowband channels in frequency domain and operating in the MF-TDMA format. We assume each terminal has only one transmitter which could possibly hop to another narrowband channel after it finishes transmitting one MAC packet in its current channel. In the above we assume the terminals are peak power constrained which puts an upper limit on the transmission rate of each narrowband channel. We further assume that the terminals are not average power constrained and they can keep on sending as long as there are packets in their MAC layer queues.

The forward channel from the hub to the terminals uses a different frequency from those used the reverse channel i.e. FDD and it is operated in a TDM fashion. Each terminal filters the received packets based on its own MAC address and only delivers those destined to it. The propagation delay between each terminal and the hub is 250ms.

4.2 Motivation

Multichannel slotted Aloha (MCSA) has been proposed by Birk and Keren to be used in the reverse channel [18]. With multiple channels, immediate retransmission following a collision is permitted by randomly choosing a channel. While in single channel slotted Aloha, temporal random retransmisson is the only option to avoid a definite repeated collision.

For short transaction based messages, it is desirable to deliver them with the smallest time delay possible. However in multichannel slotted Aloha, the success probability of a k slot message in its first attempt is e^{-G*k}, where G is the traffic

load. If we assume all the channels are operating at the same low load of $G = 0.1$ and message length equals to nine slots, the success probability of first attempts is only 0.41. This means that only 41% of the messages can be delivered in the first try i.e. with propagation delay of 250ms. While 59% of the messages will incur retransmissions and the message delay will be at least 750ms. Therefore in a GEO satellite network which has large propagation delay, the multichannel slotted Aloha cannot deliver the multislot short messages in a timely manner.

In the above the narrowband slotted Aloha is extended to wideband in the frequency domain. It is worth to mention that it is also possible to extend narrowband Aloha to wideband in the code domain. Two different approaches are proposed to combine the spread spectrum technique and Aloha together. One approach called spread Aloha [2] uses a common code for all the terminals and separates different terminals by a random timing mechanism. Another approach [29] is a slotted direct sequence spread spectrum multiple access (DS/SSMA) protocol and each terminal employs a newly chosen random signature sequence for each bit in a transmitted packet. Logically speaking, these two approaches are equivalent to the FDMA/Aloha approach in the same way of creating separate logical channels. Here we focus on the FDMA/Aloha approach and we argue that the performance of the FDMA/Aloha approach should be similar to the other two approaches.

In additional to the random access protocols, reservation based protocols are also proposed in the literature. one protocol called combined free demand assignment multiple access (CFDAMA) [60] introduces the new concept of free bandwidth assignment. CFDAMA first allocates reverse channel bandwidth to the terminals on a demand basis. However when there is no demand, the scheduler allocates the remaining free bandwidth to the terminals according to some schedul-

ing schemes such as round robin. The reverse channel bandwidth is managed by a centralized scheduler located at the hub on the ground. CFDAMA is efficient for bulk data transfers. However for bursty short transfers, due to the very large terminal population, the probability for a terminal to receive a free allocation is pretty low. Therefore most of the time a bandwidth request has to be sent which introduces additional delay of about 500ms.

From above, we can see the proposed protocols do not perform well in term of efficiency and delay when driven by bursty short messages. This motivates us to design a new MAC protocol which can deliver short messages with small delay and reasonable efficiency. Fixed assignment is ruled out of our consideration because it is not efficient for the bursty traffic. Reservation based protocols introduce the request phase overhead which is too expensive for short messages. Our protocol is a random access protocol which can improve the delay performance of multislot short messages dramatically compared with the aforementioned protocols.

4.3 Fast Multichannel Slotted Aloha

Our protocol called fast multichannel slotted Aloha (FMCSA) is based on multichannel slotted Aloha (MCSA) with new features such as packet level FEC and scheduled ARQ designed specifically for networks with large propagation delay. The system model of our protocol is shown in figure 4.1.

When a new message arrives from the upper layer, it is first segmented into k MAC packets[1]. Then the k packets are encoded into n code packets [5] [67] [82] [74] [73]. The first k code packets contain the original message. The remaining $n-k$ are parity packets. The n code packets are sent in n consecutive time slots and each

[1]We assume one slot can carry exactly one MAC packet

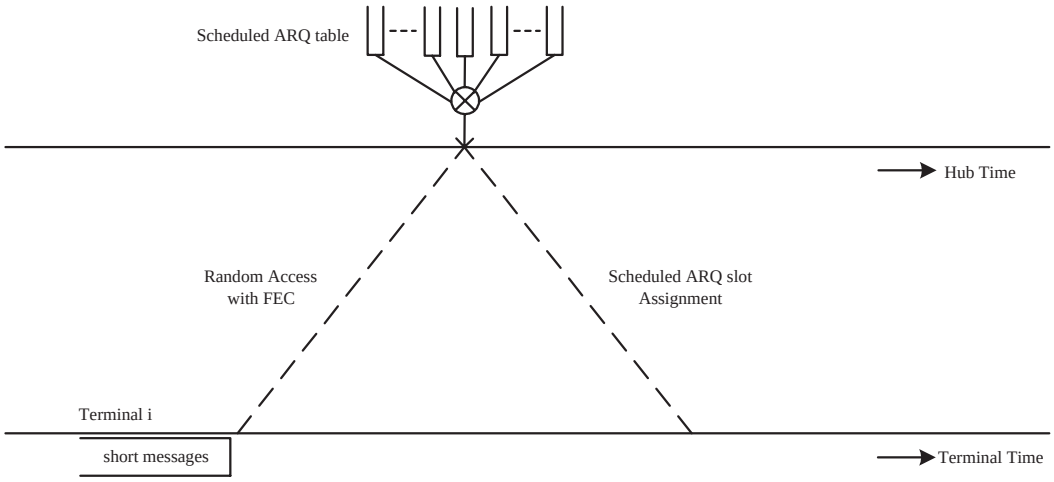


Figure 4.1: System model of fast multichannel slotted Aloha

slot is randomly chosen from the remaining FDMA Aloha channels which have not been reserved for the retransmissions. When the hub schedules the retransmissions, it makes sure that for each slot there is at least one channel left for random access. For every n code packets, if any k or more out of them are received, the original message can be recovered correctly from the erasure. Each of the n code packets carries a unique message id number and a sequence number. In order to decrease the bandwidth overhead caused by sending the acknowledgements, FMCSA does not send one acknowledgement for each MAC packet as mentioned in section 4.1 rather it sends an acknowledgement for each message. There are three possible outcomes of a message after its first attempt.

In the first case, the message is fully received. In this case, the number of packets received correctly m is no less than k so that the message can be reassembled and forwarded to the upper layer. A positive acknowledgement for this message will be sent to the terminal as soon as m becomes equal to k. It is possible that additional parity packets of the message will arrive after this. Because the whole message has already been received correctly, the additional parity packets will

simply be discarded by the hub.

In the second case where the message is partially received i.e. m is less than k but greater than zero, packet retransmission becomes necessary. A selective reject strategy [80] [81] is employed in FMCSA for packet recovery. From the sequence numbers and the message id number carried in the MAC packets, the hub can figure out which packets are collided in the message and it will reserve $k-m$ slots for the message recovery rather than let the terminal to retransmit the collided packets in the random access mode [95]. Because of the scheduled ARQ, FMCSA can guarantee the successful delivery of a message in its second attempt as long as there is at least one packet getting through in its first attempt. For example, a three-slot message is encoded into five code packets and they are sent in five consecutive slots. If only the second and the fourth code packets are received correctly, the hub will find out that the message length is three and it needs one more code packets to reassemble the message. It then allocates one slot for this message. After the terminal receives the allocation, it sends an additional code packet such as the first code packet in the reserved slot so that the hub will be able to recover the message after it receives this packet.

In the worst case where none of the n code packets get through i.e. the whole message is erased, the terminal will timeout and the collided n code packets will be sent again in the same way a new message is transmitted.

4.4 Performance Analysis

In this section, we model FMCSA and analyze its throughput and delay performance. We will show that FMCSA can achieve a higher maximum throughput than MCSA. For the same system throughput, FMCSA can deliver the short mes-

sages in the first attempts with much higher probability than MCSA. With the increasing of system load, the message delay performance is still acceptable and it degrades much more gracefully than MCSA.

4.4.1 System Throughput

The system throughput we are interested in is the throughput seen in the upper layer i.e. the effective throughput of the short messages which does not include the throughput of the redundant parity packets. In the analysis, we assume that the new MAC packet arrival rate in each of the N parallel channels is Poisson distributed with mean λ_k. Therefore after FEC coding, the arrival rate is increased to $\lambda_n = \frac{n}{k} * \lambda_k$. We classify the retransmissions into two cases. In the first case, a n slot message is partially received and scheduled ARQ is used for its retransmission. We assume that the arrival rate of scheduled ARQ assignments in each channel is Poisson distributed with mean λ_r. In the second case, none of the n MAC packets in a message is received and all of them have to be sent again. We assume its arrival rate is also Poisson distributed with mean λ_{nr}.

The system throughput S contains two parts. One is contributed by the n slot messages which include new message transmissions and totally erased message retransmissions $S(\lambda_n, \lambda_{nr}, \lambda_r)$. Note that it is also a function of the scheduled retransmission rate λ_r. This is because the n slot messages are sent in the channels left by the scheduled retransmissions. The other part of the system throughput is contributed by the less than k slot retransmissions i.e. the scheduled ARQ $S(\lambda_r)$. Because reservation is used in scheduled ARQ, it is guaranteed that all such retransmissions will be received correctly i.e.

$$S(\lambda_r) = \lambda_r \tag{4.1}$$

Because the channels reserved for scheduled ARQ cannot be used by the n slot messages, it is equivalent that their arrival rates λ_n and λ_{nr} are both increased by a factor of $1/(1-\lambda_r)$. Let the total arrival rate of n slot messages be G_n, then we have

$$G_n = \frac{\lambda_n + \lambda_{nr}}{1-\lambda_r} \tag{4.2}$$

where

$$\lambda_n = \frac{n}{k} * \lambda_k \tag{4.3}$$

It should be noted that not all the correctly received code packets contribute to the effective throughput S. Let the number of correctly received packets in a single n slot message be m. If $0 \leq m \leq k$, all the m received packets contribute to the effective throughput S. On the other hand, if $m > k$, only k out of m code packets contribute to S. The probability of m code packets received in a n slot message is

$$P_r(m) \ = \binom{n}{m} \cdot (1-e^{-G_n})^{n-m} \cdot e^{-G_n * m} \tag{4.4}$$

The probability of an arbitrary received code packet contributes to S is

$$P_e = \frac{\sum_{m=0}^{n} min(m,k) \cdot P_r(m)}{\sum_{m=0}^{n} m \cdot P_r(m)} \tag{4.5}$$

Therefore the throughput of the n slot messages is

$$S(\lambda_n, \lambda_{nr}, \lambda_r) \ = P_e \cdot (\lambda_n + \lambda_{nr}) \cdot e^{-G_n} \tag{4.6}$$

The effective system throughput is

$$S \ = S(\lambda_n, \lambda_{nr}, \lambda_r) + S(\lambda_r) \tag{4.7}$$

In steady state, the arrival rate of the new packets equals to the effective system throughput $S = \lambda_k$, then from equations 4.7, 4.6 and 4.1, we can get

$$\lambda_k = P_e \cdot (\lambda_n + \lambda_{nr}) \cdot e^{-G_n} + \lambda_r \tag{4.8}$$

Also note that the MAC packet arrival rate due to the totally erased message retransmissions is as following

$$\lambda_{nr} = (\lambda_n + \lambda_{nr}) \cdot (1 - e^{-G_n})^n \tag{4.9}$$

From equations 4.2 and 4.9, we can have

$$\lambda_n = (1 - \lambda_r) \cdot G_n \cdot (1 - (1 - e^{-G_n})^n) \tag{4.10}$$

$$\lambda_{nr} = (1 - \lambda_r) \cdot G_n \cdot (1 - e^{-G_n})^n \tag{4.11}$$

From equations 4.3 and 4.10, we can get

$$\lambda_k = \frac{k}{n} \cdot (1 - \lambda_r) \cdot G_n \cdot (1 - (1 - e^{-G_n})^n) \tag{4.12}$$

Substitute λ_k with the right side in equation 4.12 and $(\lambda_n + \lambda_{nr})$ with $(1 - \lambda_r) \cdot G_n$ into equation 4.8, we can get the following

$$\frac{k}{n} \cdot (1 - \lambda_r) \cdot G_n \cdot (1 - (1 - e^{-G_n})^n) = P_e \cdot (1 - \lambda_r) \cdot G_n \cdot e^{-G_n} + \lambda_r \tag{4.13}$$

Equation 4.13 relates G_n and λ_r, G_n can be calculated numerically for a given λ_r. Once G_n and λ_r are known, we can get the system throughput λ_k by using equation 4.12 as well as the n slot message retransmission rate λ_{nr} through equation 4.9in which λ_n equals $\frac{n}{k} * \lambda_k$.

Figure 4.2 shows the throughput performance of FMCSA and MCSA with respect to the total offered load G. In FMCSA, $G = \lambda_n + \lambda_{nr} + \lambda_r$ where λ_n is the new code packet arrival rate. We can see from the figure that FMCSA can achieve

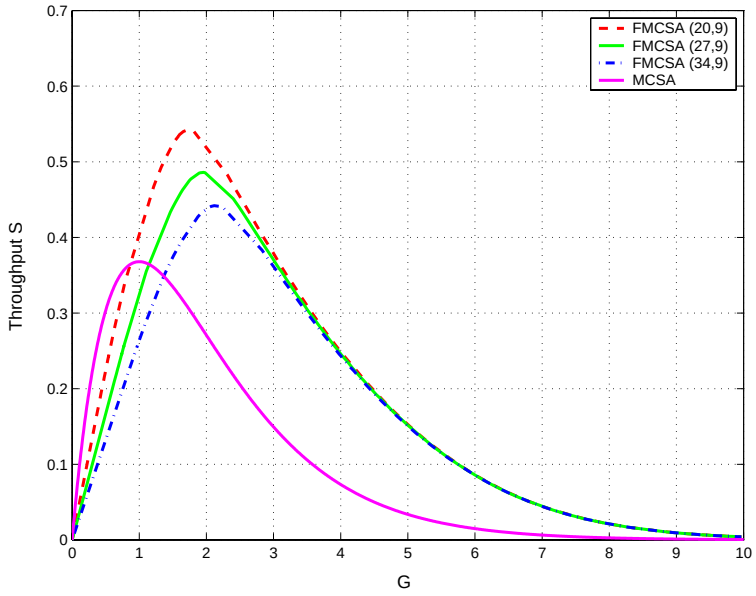


Figure 4.2: The system throughput of FMCSA and MCSA

a higher maximum throughput than MCSA. The maximum throughput of MCSA is 0.368. While FMCSA with a FEC code (34,9) has a maximum throughput of 0.442. With the code rate increased to 9/27, the maximum throughput is increased to 0.486. When FEC code (20,9) is used, the maximum throughput becomes 0.542. FMCSA have the same bistable property as MCSA. We would like to operate the two protocols in the stable region, i.e. on the left side of the maximum throughput point. Due to the parity packets sent in FMCSA, more load is offered to the channel than in MCSA to achieve the same throughput. However the figure does not give too much information about the message delay. In the next section will compare the delay performance of FMCSA and MCSA for a given throughput.

4.4.2 First Attempt Success Probability

Because the propagation delay in satellite networks is very large, it is desirable that the messages can be received correctly in the first attempts. Otherwise one

round trip time delay about 500ms will be introduced.

In the following, we calculate the first attempt success probability of FMCSA with a FEC code (n, k). MCSA is equivalent to the degenerated case of FMCSA with $n = k$. Let p be the probability with which a new MAC packet can get through the channel. Therefore,

$$p = e^{-G} \; in \; MCSA \tag{4.14}$$

$$p = e^{-G_n} \; in \; FMCSA \tag{4.15}$$

Where G is the total offered load in the MCSA channel and G_n is given by equation 4.2 which is the load offered to the channel left by the scheduled ARQ in FMCSA.

The full message received probability, i.e. the probability of no less than k packets received in the first attempt is

$$P_r(m \geq k) \; = \sum_{m=k}^{n} \binom{n}{m} \cdot (1-p)^{n-m} \cdot p^m \tag{4.16}$$

The probability of none of the n packets getting through is

$$P_r(m = 0) \; = (1-p)^n \tag{4.17}$$

The probability of partially received messages in the first attempts, i.e. $0 < m < k$ is

$$P_r(0 < m < k) \; = 1 - P_r(m \geq k) - P_r(m = 0) \tag{4.18}$$

Numerical results are shown in table 4.1 and table 4.2 about the probability of the above three cases for different system throughput in MCSA and FMCSA. Here $k = 9$ and a (27,9) error correction code is used ($n = 27$) in FMCSA. From the tables, we can see that the successful probability in the first attempt decreases

very fast in MCSA with the increase of the throughput. While in FMCSA, this probability is not sensitive to the throughput. Actually when the system throughput S is below 0.25, almost all the messages can get through in their first attempts. Even when the throughput is increased up to 0.3, the successful probability of a message in its first attempt is as high as 82.31% in FMCSA compared with only 1.22% in MCSA. Table 4.3 and table 4.4 show similar results for FEC code (20,9) and (34,9) respectively.

Figure 4.3 and Figure 4.4 show the distribution of the number of packets that get through in the first attempt for different system throughput in MCSA and FMCSA respectively. In MCSA the probability of all the nine MAC packets received successfully decreases significantly with the increase of the load. While in FMCSA the probability of more than k MAC packets are received successfully decreases much more gracefully with the increase of the throughput. At the same time, we notice that the probability of none of the n code packets[2] getting through is pretty low for both protocols even when the throughput is high. This means almost all the messages are either fully received or partially received in the first attempt. Because the packets of a partially received message can reserve enough slots to retransmit the additional code packets in FMCSA, the message can be recovered in the second attempt. While MCSA does not take advantage of this fact, the retransmissions may again incur collisions. Therefore FMCSA can improve the delay performance significantly when compared with MCSA. Figure 4.5 and figure 4.6 shows similar distributions for FEC code (20,9) and (34,9) respectively. From above, we can see that the first attempt success probability in FMCSA is not sensitive to system throughput as well as FEC code rate.

[2] $n = 27, k = 9$ in FMCSA and $n = 9, k = 9$ in MCSA

Table 4.1: None, partial and full message received probability in MCSA with degenerated FEC code (9,9)

S	$P_r(m = 0)$	$P_r(0 < m < k)$	$P_r(m \geq k)$
0.01	$1.045 * 10^{-18}$	0.0869	0.9131
0.05	$2.476 * 10^{-12}$	0.3777	0.6223
0.1	$1.658 * 10^{-9}$	0.6344	0.3656
0.15	$8.730 * 10^{-8}$	0.8012	0.1988
0.2	$1.687 * 10^{-6}$	0.9030	0.0970
0.25	$1.998 * 10^{-5}$	0.9599	0.0401
0.3	$1.947 * 10^{-4}$	0.9876	0.0122
0.35	$2.400 * 10^{-3}$	0.9960	0.0016
0.4	N/A	N/A	N/A

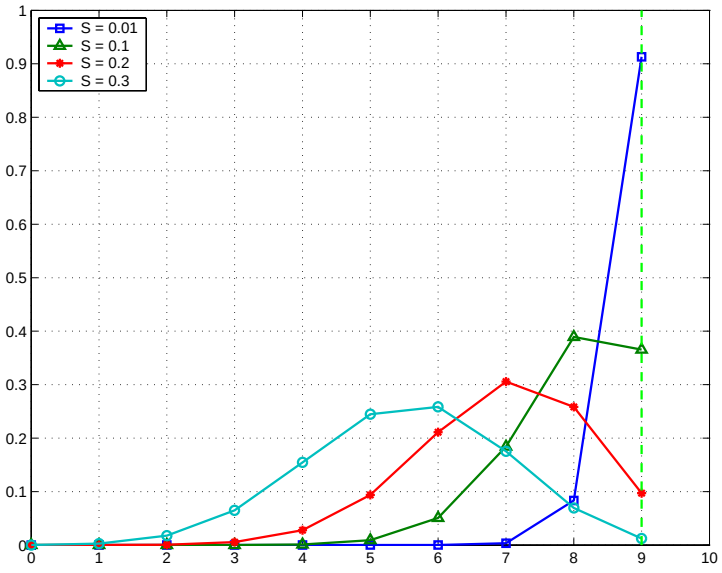


Figure 4.3: The distribution of number of received packets in MCSA with degenerated FEC code (9,9)

Table 4.2: None, partial and full message received probability in FMCSA with FEC code (27,9)

S	$P_r(m = 0)$	$P_r(0 < m < k)$	$P_r(m \geq k)$
0.01	$\simeq 0$	$\simeq 0$	$\simeq 1$
0.05	$\simeq 0$	$\simeq 0$	$\simeq 1$
0.1	$1.470 * 10^{-16}$	$1.684 * 10^{-6}$	0.9999
0.15	$1.251 * 10^{-12}$	$3.261 * 10^{-4}$	0.9997
0.2	$4.652 * 10^{-10}$	$7.144 * 10^{-3}$	0.9929
0.25	$3.184 * 10^{-8}$	$4.896 * 10^{-2}$	0.9510
0.3	$9.134 * 10^{-7}$	$1.769 * 10^{-1}$	0.8231
0.35	$1.566 * 10^{-5}$	$4.156 * 10^{-1}$	0.5844
0.4	$3.029 * 10^{-4}$	$7.509 * 10^{-1}$	0.2488

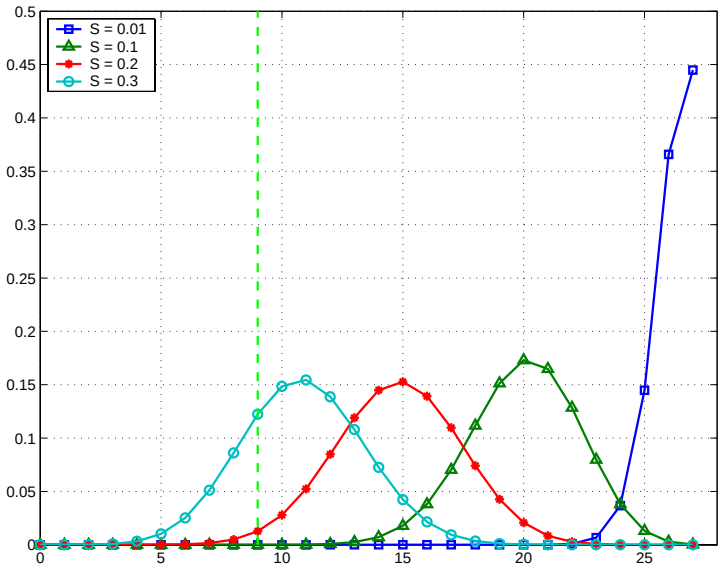


Figure 4.4: The distribution of number of received packets in FMCSA with FEC code (27,9)

Table 4.3: None, partial and full message received probability in FMCSA with FEC code (20,9)

S	$P_r(m = 0)$	$P_r(0 < m < k)$	$P_r(m \geq k)$
0.01	$\simeq 0$	$\simeq 0$	$\simeq 1$
0.05	$\simeq 0$	$\simeq 0$	$\simeq 1$
0.1	$9.800 * 10^{-15}$	$9.827 * 10^{-5}$	0.9999
0.15	$1.126 * 10^{-11}$	$3.068 * 10^{-3}$	0.9969
0.2	$1.268 * 10^{-9}$	$2.417 * 10^{-2}$	0.9758
0.25	$4.174 * 10^{-8}$	$9.203 * 10^{-2}$	0.9080
0.3	$6.954 * 10^{-7}$	$2.297 * 10^{-1}$	0.7703
0.35	$8.517 * 10^{-6}$	$4.423 * 10^{-1}$	0.5577
0.4	$1.106 * 10^{-4}$	$7.129 * 10^{-1}$	0.2870

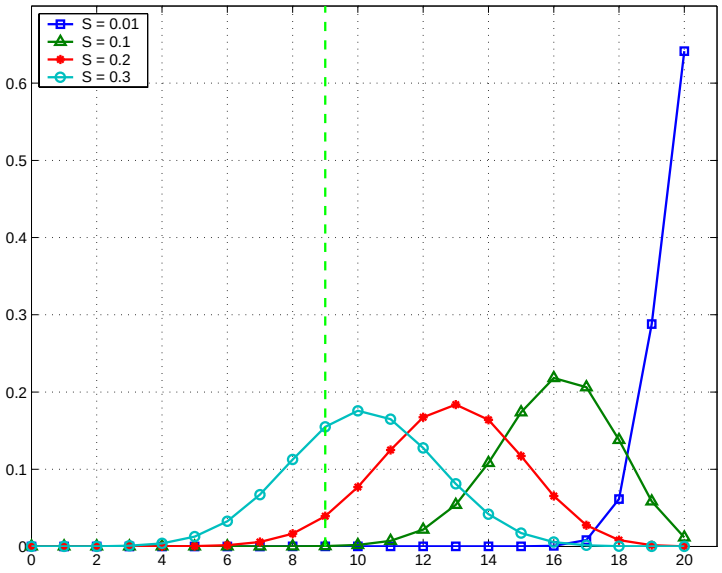


Figure 4.5: The distribution of number of received packets in FMCSA with FEC code (20,9)

Table 4.4: None, partial and full message received probability in FMCSA with FEC code (34,9)

S	$P_r(m = 0)$	$P_r(0 < m < k)$	$P_r(m \geq k)$
0.01	$\simeq 0$	$\simeq 0$	$\simeq 1$
0.05	$\simeq 0$	$\simeq 0$	$\simeq 1$
0.1	$8.546 * 10^{-18}$	$9.026 * 10^{-8}$	0.9999
0.15	$4.257 * 10^{-13}$	$8.625 * 10^{-5}$	0.9999
0.2	$4.308 * 10^{-10}$	$4.311 * 10^{-3}$	0.9957
0.25	$5.575 * 10^{-8}$	$4.593 * 10^{-2}$	0.9541
0.3	$2.612 * 10^{-6}$	$2.103 * 10^{-1}$	0.7897
0.35	$6.095 * 10^{-5}$	$5.224 * 10^{-1}$	0.4775
0.4	$2.148 * 10^{-3}$	$9.014 * 10^{-1}$	0.0965

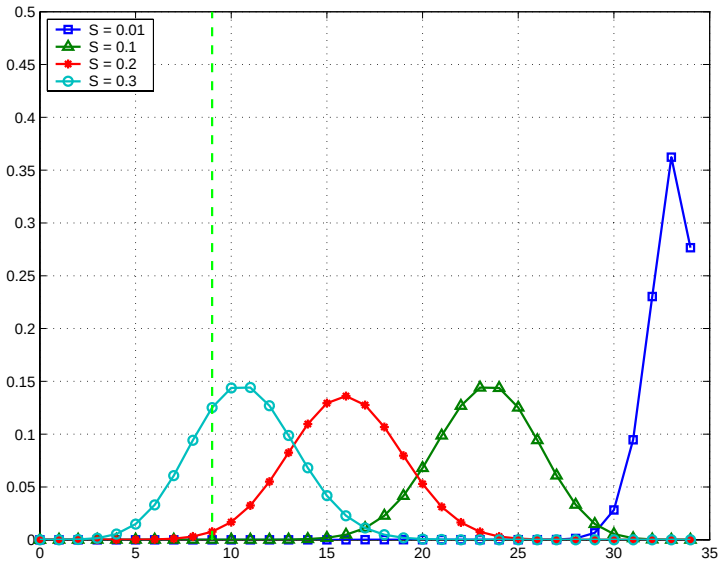


Figure 4.6: The distribution of number of received packets in FMCSA with FEC code (34,9)

4.4.3 Retransmission Rate and Operation Region

In MCSA, the average number of attempts for the successful reception of a single MAC packet is e^G. Therefore the average delay for MAC packet i denoted by $D_{MCSA}(i)$ [54] is given as follows

$$D_{MCSA}(i) = 1 + t_{prop} + (e^G - 1)(1 + 2 * t_{prop}) \tag{4.19}$$

Where t_{prop} is the normalized propagation delay[3]. Because the t_{prop} is large in satellite networks, from equation 4.19 we can see that retransmissions will increase the packet delay dramatically as the load G increases. Therefore to deliver the packets with short delay, the MCSA channel bandwidth should be sized such that it operates in the low load region such as $G = 0.1$ most of the time, in which every attempt including first time transmissions and retransmissions of a packet has a probability of 90.5% to get through the channel.

While in the above we calculate the delay for a single MAC packet, the whole k slot message delay is determined by the delay of the last received packet generated from the message. Therefore the message delay in MCSA is

$$D_{MCSA} = max(D_{MCSA}(1), D_{MCSA}(2), ..., D_{MCSA}(k)) \tag{4.20}$$

Even operating in the low load region such as $G = 0.1$, the probability of the message getting through in the first attempt is only 41% as calculated in section 4.2. This means to improve the message delay performance in MCSA, the channel may have to be operated in an even lower load region than $G = 0.1$, which of course is very inefficient in term of bandwidth utilization.

In the following we will show that FMCSA can operate much more efficiently without sacrificing the delay performance. The retransmissions in FMCSA include

[3]The propagation delay t_{prop} is normalized to the packet transmission time

two parts: the scheduled ARQ and the totally erased message retransmissons. We plot these two retransmission rates with respect to the system throughput in figure 4.7, figure 4.9 and figure 4.10 for different code rates. Our discussion will focus on the $(27, 9)$ code. The other two codes perform similarly to this code. Figure 4.7 shows the two retransmission rates at different operating points. The curve can be divided into two regions i.e. the stable region[4] and the unstable region. In the unstable region, we can see that both retransmission rates are relatively large.

We would like to have FMCSA operate in the stable region. Figure 4.8 shows a more detailed n slot retransmission rate in the stable region. When the system throughput is less than 0.2, the scheduled ARQ rate λ_r and the n slot retransmission rate are both pretty close to zero. This means the messages can get through in their first attempts which confirms the results we get in table 4.2. The delay performance of FMCSA is almost the same in region below $S = 0.2$ and is much more robust to system load than that of MCSA.

When the system throughput S is between 0.2 and 0.4, we can see from the figure 4.8 that λ_{nr} is still very small and the scheduled ARQ rate begins to increase as shown in the upper plot of figure 4.7. This plot shows that as the system throughput increases, more messages will be delivered in the second attempts. From the above we can see that FMCSA should operate below throughput of 0.2 to get small message delay. While even under temporary congestion, FMCSA can still be functional with acceptable message delay.

[4]In the upper plot of figure 4.7, the stable region is the part below the line connecting $(0, 0)$ and (S_{max}, λ_r^*). In the lower plot of figure 4.7, the stable region is the part below the line connecting $(0, 0)$ and $(S_{max}, \lambda_{nr}^*)$.

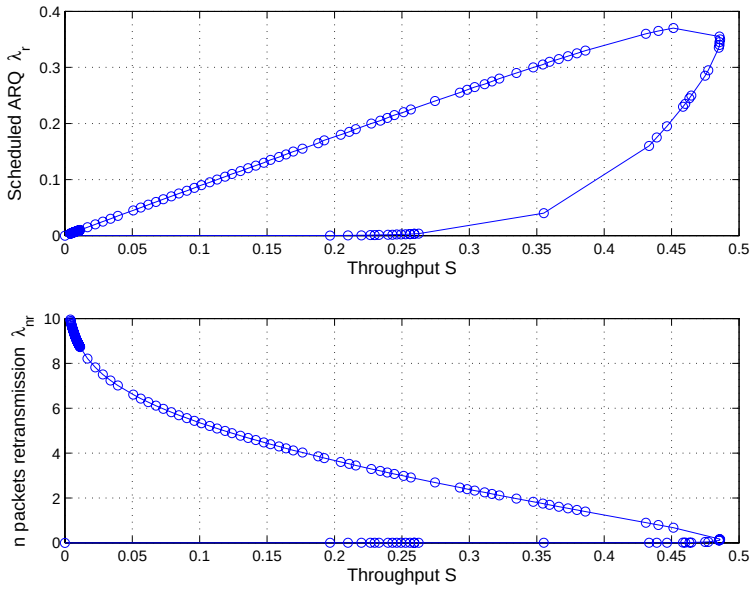


Figure 4.7: The retransmission rate of FMCSA for different system throughput with FEC code (27,9)

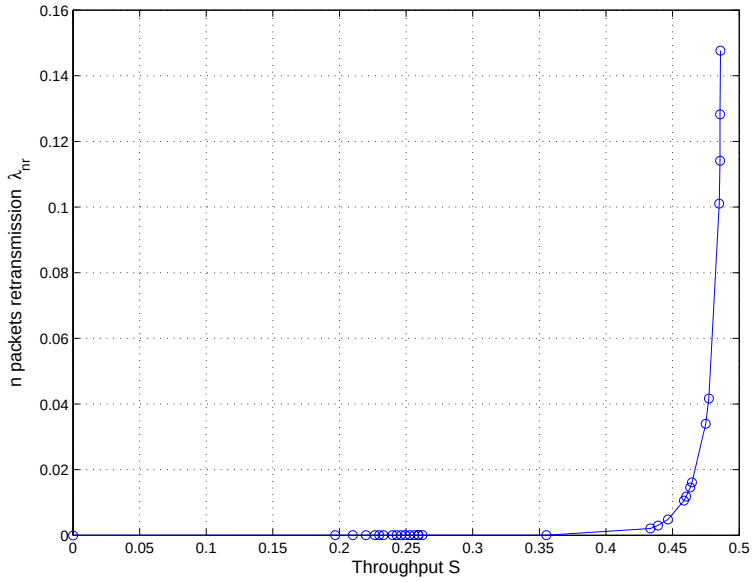


Figure 4.8: The totally erased message retransmission rate of FMCSA for different system throughput with FEC code (27,9)

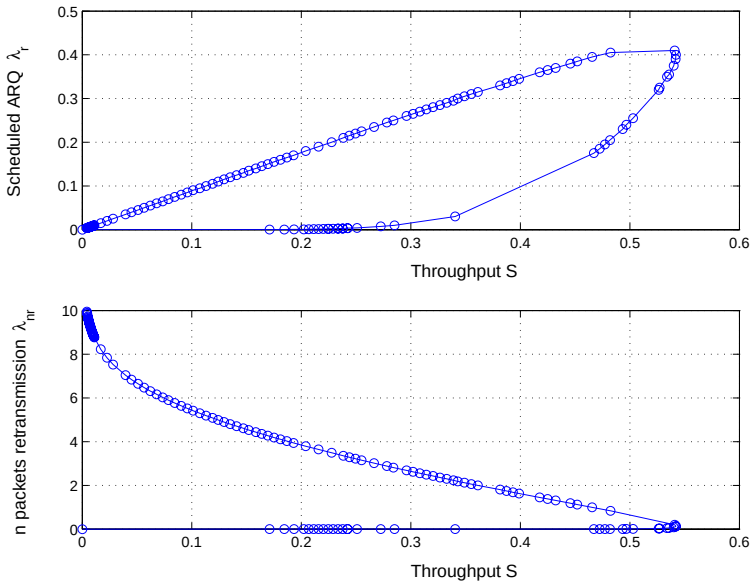


Figure 4.9: The retransmission rate of FMCSA for different system throughput with FEC code (20,9)

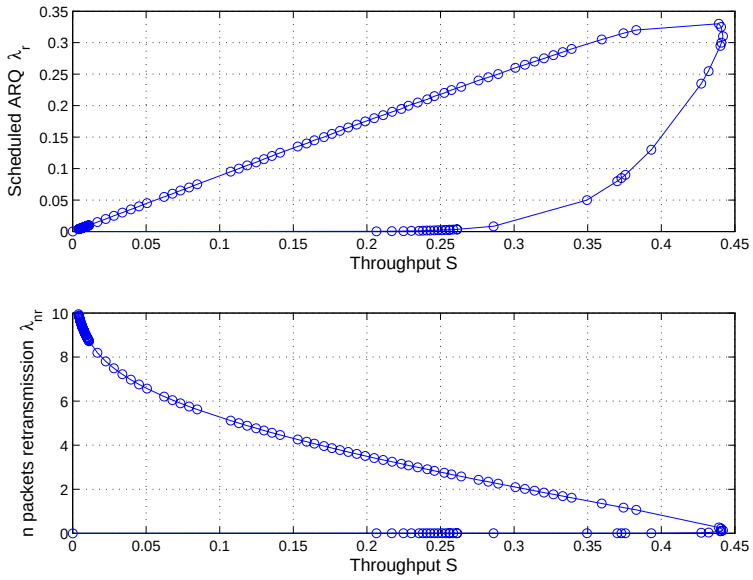


Figure 4.10: The retransmission rate of FMCSA for different system throughput with FEC code (34,9)

4.5 Simulation Results

In this section, we evaluate the delay performance of FMCSA and MCSA in OPNET. We will show that the simulation results with Poisson arrivals match very well with the the analytical results we get in the previous section. We further evaluate the performance of FMCSA with Pareto arrivals. Our results show that the delay performance of FMCSA is not sensitive to the specific arrival distributions when the reverse channel is operating in the relatively low load region.

4.5.1 Delay Performance with Poisson Arrivals

In this section, we evaluate the delay performance of FMCSA with Poisson arrivals. There are 512 terminals in the network and all of them are sending short messages to the hub. The MAC packet size is 53 bytes with 5 bytes header. All the messages generated by the upper layer are of 9 slots i.e. 432 bytes. There are 25 parallel reverse channels and each has bandwidth of 64 kbps. Figure 4.11 shows the average message delay of FMCSA and MCSA for different throughput. The minimum message delay is achieved when the first 9 MAC packets of a message are received correctly at the hub and the delay is (250 + 9*53*8/64) i.e 309.625 ms which is shown as the dash line in figure 4.11. We can see from the figure that when the system throughput is very small at around 1%, the average delay of both schemes are close to the minimum delay. While with the increase of the throughput, the average delay of MCSA increase dramatically. When the throughput is increase up to 35%, MCSA actually becomes saturated and the delay goes to infinity. As predicted by our performance analysis of FMCSA, its delay performance degrades much more gracefully than MCSA. Even when the system throughput is increased up to 20%, the average delay performance is comparable to the minimum delay.

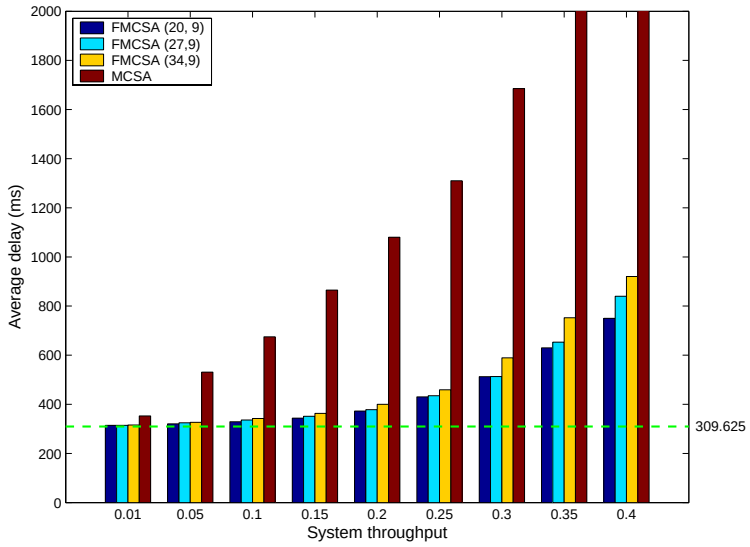


Figure 4.11: The average message delay of FMCSA and MCSA for different throughput. There are 512 terminals and 25 parallel reverse channels. Each channel has bandwidth of 64kbps.

Figure 4.11 also shows that FMCSA can achieve a higher throughput than MCSA which makes it more robust to load fluctuation. For different code rate, there is not too much difference for the average delay when operating at the relatively low load region such as throughput less that 25%. In practice, this property gives us the flexibility to choose different codes.

Figure 4.12 shows the instantaneous message delay performance of FMCSA with FEC code (27,9) for three different throughput. If a new message arrives at an empty MAC layer queue and it is received correctly in the first attempt, the minimum delay is 309.625 ms as calculated in the above paragraph and similarly we can get the maximum delay is (250 + 27*53*8/64) i.e. 428.875 ms. Therefore if a message is not received fully in its first attempt, the message delay should be more than (428.875 + 2*250) i.e. 928.875ms. So if the delay of a message is less than 928.875 ms, it must has been received in the first attempt. In topmost plot of figure

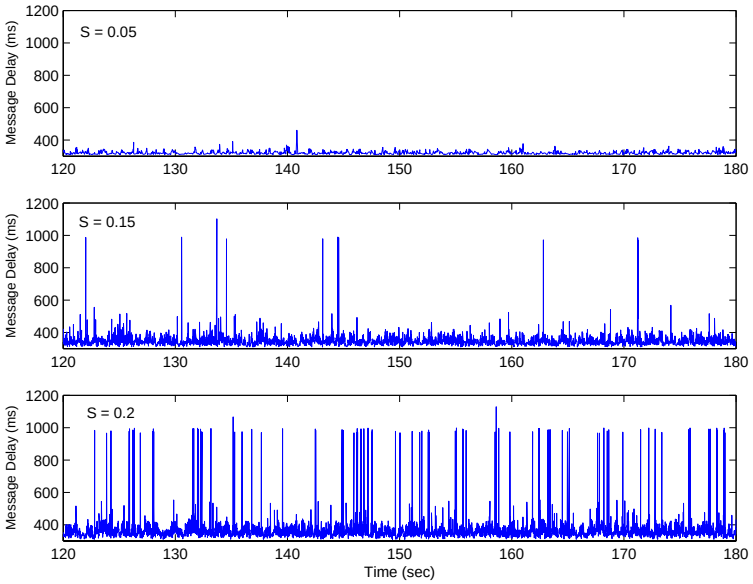


Figure 4.12: The message delay of FMCSA (27,9) for different throughput. There are 512 terminals and 25 parallel reverse channels. Each channel has bandwidth of 64kbps.

4.12, we can see that the messages are delivered in the first attempts when the throughput is 5%. When the throughput is increased to 15%, very few of them will incur retransmissions. We should point out here that because we allow multiple messages to be buffered at the terminals. It could happen that a new message arrives at the MAC layer queue and the terminal has not finished transmitting the previous packets. Under such circumstance even if the new message can be received in its first attempt, its message delay could be more than 428.875 ms, which corresponds to the case when a message arrives at an empty MAC queue thus without any additional queuing delay[5]. This is confirmed by the middle plot in figure 4.12. When the throughput is increased further, we can see more message

[5]Because a packet can only be sent at the beginning of a slot, the packet may still incur some queuing delay of less than one slot

Table 4.5: The analytical and simulation results of the first attempt success probability in FMCSA with a FEC code (27,9)

S	0.05	0.1	0.15	0.2	0.25	0.3	0.35	0.4
Analysis	$\simeq 1$	0.9999	0.9997	0.9929	0.9510	0.8231	0.5844	0.2488
Simulation	$\simeq 1$	$\simeq 1$	0.9979	0.9767	0.9068	0.7875	0.5674	0.2735

will have to be retransmitted in the second attempts. We also calculate the first attempt success probabilities and compare them with our analysis in section 4.4.2. As shown in table 4.5, the simulation results match the analytical results very well.

We also evaluate the delay performance of FMCSA and MCSA with more terminals and higher bandwidth. Figure 4.13 shows the average delay of FMCSA and MCSA with 1024 terminals and 50 parallel reverse channels. Each channel has a bandwidth of 128kbps, therefore the total reverse channel bandwidth is 6.4Mbps. Because of the increase of the bandwidth, the transmission time of a 9 slot message is decreased to (9*53*8/128) i.e. 29.8125ms. In both FMCSA and MCSA, the minimum message delay is achieved when all of the first 9 slot MAC packets are received correctly in the first attempt and it is the transmission time plus the propagation delay (i.e. 29.8125+250 = 279.8125ms) as shown in figure 4.13. Compared with the minimum delay of 309.625ms when each channel bandwidth is 64kbps, the increase of the bandwidth reduces the minimum delay by 29.8125ms. We can see in figure 4.13 that the average delay of FMCSA is very close to the minimum delay when the throughput is relatively small. With the increase of throughput, the delay performance of FMCSA degrades much more gracefully than that of MCSA. Actually figure 4.13 and figure 4.11 shows very similar results

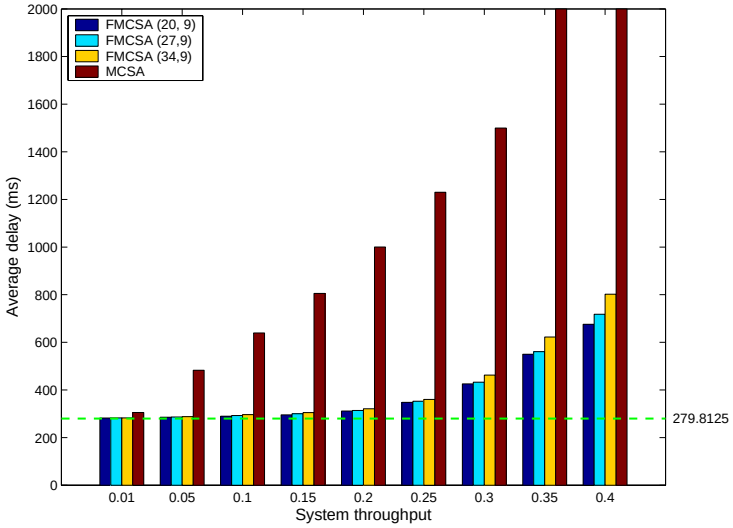


Figure 4.13: The average message delay of FMCSA and MCSA for different throughput. There are 1024 terminals and 50 parallel reverse channels. Each channel has bandwidth of 128kbps.

which gives us the evidence that FMCSA can scale to larger networks and higher channel bandwidth.

From above, we can see that FMCSA provides a system designer two dimensions of freedom to add more terminals to the network while keeping the delay performance the same. The network dimensioning can be done simply by adding more parallel channels while remaining the channel bandwidth as before. Another option is to leave the total number of channels unchanged however increase the bandwidth of each channel. As mentioned before, there is a limitation of each channel's peak power so that its bandwidth should not exceed some threshold. Therefore under some circumstance it requires the third design option which increases the number of channels and the bandwidth of each channel at the same time as we do in this experiment.

4.5.2 Delay Performance with Pareto Arrivals

In the previous section, the message arrival rate follows the Poisson distribution. In this section, we evaluate the performance of FMCSA when the message arrival pattern is Pareto distributed [28] [72]. Pareto distribution is the simplest heavy-tailed distribution with probability density function

$$p(x) = \alpha k^{\alpha} x^{-\alpha-1} \quad where\ \alpha, k > 0, x \geq k$$

and cumulative distribution function

$$F(x) = P[X \leq x] = 1 - (k/x)^{\alpha}$$

The parameter k is the location parameter and it represents the possible smallest value of random variable X. If $\alpha \leq 2$, it has infinite variance; if $\alpha \leq 1$, it has infinite mean. For $1 < \alpha < 2$, the mean of Pareto distribution is $\alpha/(\alpha - 1) * k$. The Pareto distribution is hyperbolic over the entire range and as α decreases, an arbitrary large portion of the probability mass could be present in the tail of the distribution [28]. Therefore a Pareto distributed random variable can generate extremely large values with nonnegligible probability.

Figure 4.14 shows the cumulative distribution of the message delay for Poisson and Pareto arrivals. There are 512 terminals and 25 parallel reverse channels in the network. Each channel has a bandwidth of 64kbps and FEC code (27,9) is used. In the Pareto distribution, α equals 1.5. Figure 4.14 shows that when the throughput is up to 20%, the delay distributions for both Poisson and Pareto arrivals are pretty close to each other. While the throughput is increased above 25%, the mean delay of Pareto arrivals is higher than Poisson arrivals due to the greater burstiness of the traffic. From this figure, we can see that the delay performance of FMCSA is robust to the arrival distributions when it is operated at the relatively low load

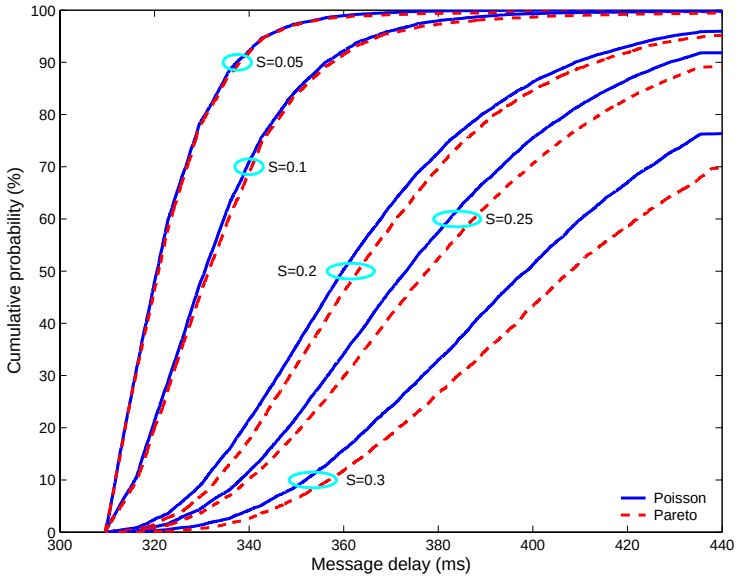


Figure 4.14: The cumulative distribution of message delay in FMCSA (27, 9) with Poisson and Pareto arrivals. There are 512 terminals and 25 parallel reverse channels in the network. Each channel has bandwidth of 64kbps.

region, which allows us to use the more mathematically tractable Poisson traffic model to predict the system performance.

4.6 Deployment Considerations

In this section, we will consider how FMCSA performs for the different system parameters such as different slot sizes and how FMCSA can be extended to handle heterogeneous message lengths. Finally, we will discuss the processing and bandwidth overhead introduced by FMCSA.

4.6.1 Handling Homogeneous Message Lengths with Different Slot Sizes

In the previous sections, we evaluate the performance of FMCSA for nine-slot messages. However when either the message length changes or a different slot size has been chosen, the number of MAC packets after the fragmentation of a message will change correspondingly. For example, in the previous sections, the message length is assumed to be 432 bytes and each slot can carry 48 byte payload. However if a larger slot size is chosen such that each slot can carry 72 byte payload, a 432-byte message will be fragmented into six MAC packets. On the other hand, if a smaller slot which can carry only 36 byte payload is used, a typical 432-byte message will generate twelve MAC packets after fragmentation. Figure 4.15 and figure 4.16 show the throughput and retransmission rates for six-slot and twelve-slot messages. From the topmost plots of both figures, we can see that FMCSA can achieve higher maximum throughput than MCSA for both cases. For the six-slot messages, FEC code (18,6) is used and for the twelve-slot messages, FEC code (36,12) is used. Actually the maximum throughput is 0.46 for FMCSA (18,6) and is 0.5 for FMCSA (36,12). The throughput results we show here is similar to the results in figure 4.2 which shows the throughput performance of FMCSA for nine-slot messages.

From the middle and the lowest plots in figure 4.15 and figure 4.16, we can see that the retransmisson rates are close to zero when the system throughput is less that 20% which means that almost all the messages can get through in their first attempts. This property is also very similar to the nine-slot message case. Therefore these results show that the performance of FMCSA is robust to the system parameters such as message length and slot size.

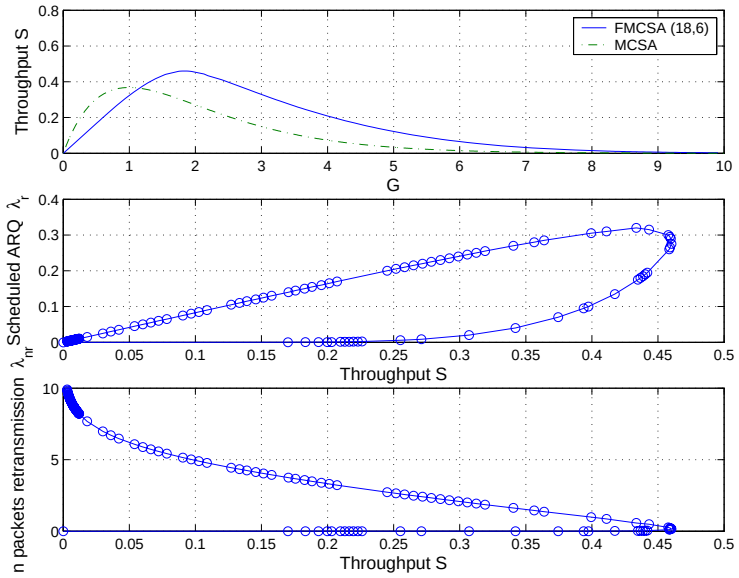


Figure 4.15: The throughput and retransmission rates of FMCSA with FEC code (18,6).

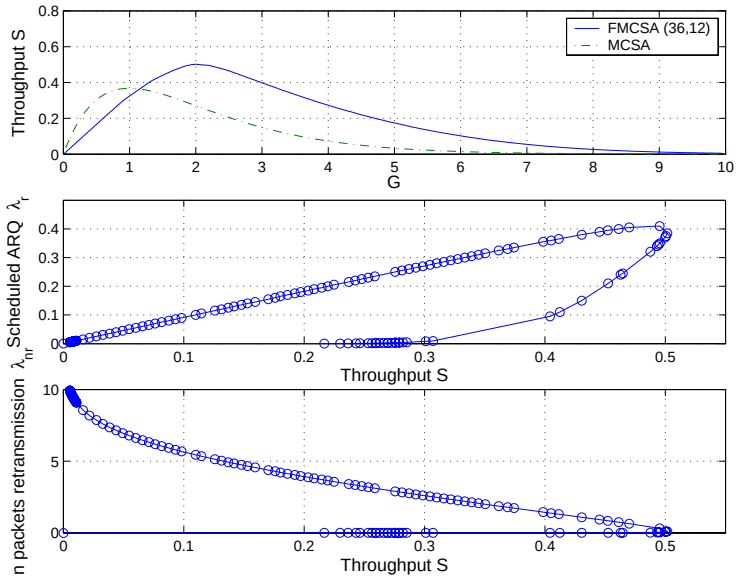


Figure 4.16: The throughput and retransmission rates of FMCSA with FEC code (36,12).

4.6.2 Handling Heterogeneous Message Lengths

In the previous sections, we evaluate the performance of FMCSA when all the messages have the same length. In practice, the message length could be heterogeneous and follow certain distributions [65]. For example, if the message lengths are distributed uniformly in the region from 384 bytes to 480 bytes and each slot can carry 48 byte payload, the number of MAC packets after fragmentation will be either eight, nine or ten. Although the message lengths are different, we can still use the same FEC code such as (30,10). In case the message is less than 10 slots, the FEC code can be shortened by conceptually making a number of zero data symbols at the encoder. The zero symbols will not be transmitted, however they will be re-inserted at the hub for the decoding. For example, for a 8-slot message, the encoder can conceptually adds two zero slots and generates a (30,10) codeword. However it transmits only the original eight data packets and certain number of parity packets.

Figure 4.17 shows the delay performance of FMCSA and MCSA for heterogeneous message length. In the upper plot, the message lengths are randomly chosen from eight, nine and ten slots with the same probability. As mentioned in the previous paragraph, the same FEC code (30,10) is used for all the messages and the number of additional parity packets is 16, 18 and 20 for 8, 9, 10 slot messages respectively. We can see from this plot that FMCSA outperforms MCSA even when the message length is heterogeneous. The upper plot also shows the delay performance of FMCSA when all the messages have the same length of nine slots. We can see from this plot that the average delay of FMCSA with heterogeneous message length is close to the homogeneous case. The lower plot shows the results for a larger message length range which is uniformly distributed from 7 to 11 slots.

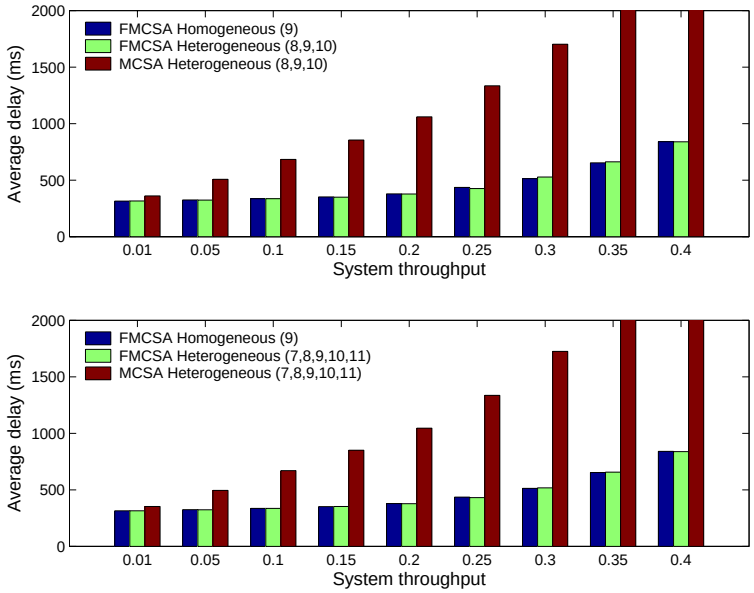


Figure 4.17: The average message delay of FMCSA and MCSA for heterogeneous message length. There are 512 terminals and 25 parallel reverse channels. Each channel has bandwidth of 64kbps.

If each slot can carry 48 byte payload, the message length is in the range from 336 bytes to 528 bytes. It shows similar delay performance of FMCSA as the upper plot in which the message lengths are distributed in a smaller range.

4.6.3 Processing and Bandwidth Overhead

In FMCSA, we introduce packet level FEC and scheduled ARQ. Compared to MCSA, FMCSA requires more CPU processing to do the FEC coding/decoding and consumes more forward channel bandwidth to send the scheduled ARQ assignments. Because the forward channel bandwidth is much larger than the reverse channel bandwidth, we consider increasing the reverse channel utilization as more important and the bandwidth consumed by the scheduled ARQ assignments is negligible compared to the forward channel total bandwidth.

A Reed-Solomon erasure (RSE) correcting code as described in [67] [82] can be used to generate the parity packets. Each terminal employs an (n, k) RSE code over a Galois field $GF(2^m)$. The symbol size m is picked to be sufficiently large such that $n < 2^m$ [74]. However, it is difficult to implement a RSE coder to operate on symbols of the message size which is typically be on the order of several throusand bits. Let the message length be $l \cdot m$ bits, where l is an integer. A multiple parallel RSE coding can be performed for each m-bit symbol in each data packet [74]. For example, RSE coding is performed on the first m-bit symbol in each of the k data packets such that $n-k$ m-bit parity symbols can be obtained. This process is then repeated for the rest $l-1$ symbols in each data packets to obtain the $n-k$ parity packets.

In the n code packets, the first k packets are generated from the segmentation of the original message. If all of the k data packets are received, no decoding is required at the receiver. On the other hand, if $j < n-k$ out of the k data packets are lost, the decoding overhead is proportional to j [74]. Nonnenmacher [74] evaluated the throughput of a software RSE codec developed by Rizzo [82] on a Pentium PC 133. It was shown that the coding and decoding throughput are on the order of 10 Mbps for $k = 7$, which is enough for our purposes. Therefore with more powerful machines and more efficient codec algorithms, the processing of RSE FEC in FMCSA should not become the system bottleneck.

4.7 Summary

In order to improve the delay performance of multislot messages in a multiple access channel with long propagation delay, a multichannel random access protocol called fast multichannel slotted Aloha (FMCSA) is proposed in this chapter. FMCSA

combines random access with packet level FEC for new messages and scheduled retransmissions for partially received messages. Through analysis and simulations, we show that FMCSA can achieve higher throughput and lower delay than MCSA. When the system is operating at relatively low load region, the short messages can be delivered in their first attempts with very high probability. We also show that the improved performance of FMCSA compared to MCSA is robust to the FEC code rate, channel bandwidth, terminal population, arrival patterns, slot size as well as message length.

In this chapter, we assume all the channels has the same bandwidth and the traffic generated by all the terminals is statistically indistinguishable from each other. We further assume that each terminal is equipped with only one transmitter and all the transmitters send packets with the same power. In the future, we would like to explore the case where the traffic loads at the terminals are heterogeneous, which happens when some terminals are used to connect a small local area network while others are used to connect a single personal computer. It is desirable to assign more bandwidth to those heavier loaded terminals either by increasing the transmission power [68] [63] or by allowing them to use more than one transmitters to send packets in parallel.

In the next chapter, we develop a closed queuing network model to capture interactions between the forward channel and the reverse channel. A systematic approach is used to evaluate web browsing performance in satellite access networks which takes into consideration the effects from application layer down to the MAC layer.

Chapter 5

Interactive Data Services in Wireless Access Networks: Capacity Planning and Protocols

In this chapter, we study the capacity planning in wireless access network for interactive data services such as web browsing. A closed queuing model has been developed which can capture the bottleneck effects in both the forward and the reverse channels. The model can be used to calculate the average throughput, the average response time and the number of users the system can support. We evaluate the performance of several MAC protocols such as slotted Aloha, static TDMA, Aloha/periodic stream and combined free demand assignment multiple access (CFDAMA) using realistic web traffic models. Based on the performance evaluation, we propose a new MAC protocol and a new transport layer protocol. Our new MAC protocol called combined polling free demand assignment multiple access (CPFDAMA) explores the correlation between forward channel data packets and reverse channel acknowledgement packets. Our new transport layer protocol

called RWBP uses per-flow queuing, round robin scheduling and receiver window backpressure for congestion management. RWBP can eliminate congestion losses inside the wireless networks. Our protocol suite outperforms the proposed protocols in term of both channel utilization and response time. Our results can be used for service providers to dimension their networks and provide quality of service to a certain number of users.

5.1 Introduction

Interactive data services such as web browsing are becoming an indispensable means for people to retrieve information from the Internet. Compared with wireline networks, wireless access networks can provide tetherless services to users even when they are on the move which creates significant opportunities for the service providers. From the providers' perspective, a major concern is how much capacity is needed to provide a certain quality of service to a certain number of users.

In this chapter, we assume that the forward channel from the hub to the terminals uses a different frequency from that used the reverse channel i.e. FDD. The forward channel is operated in a TDM fashion and the reverse channel is managed by a multiple access protocol. All the terminals and the hub are synchronized and MAC packets are allowed to be sent only at the beginning of a time slot. We assume there are no other errors except the collisions caused by the more than two packets sent in the same slot through the reverse channel. Whenever there is a collision, all packets involved are destroyed and the users have to retransmit the collided packets which introduces additional delay to the requests.

A typical user's behavior is as following: 1) The user starts a session by sending a request; 2) The request gets through the reverse channel; 3) The hub begins

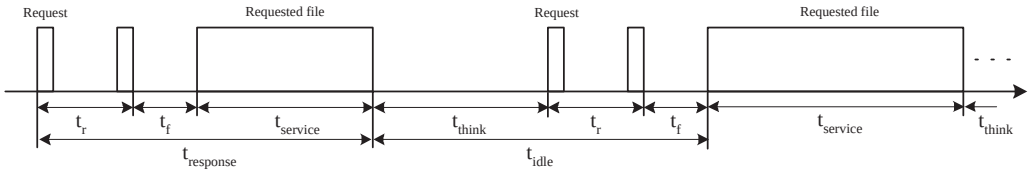


Figure 5.1: The interactive user behavior model

to send the requested file in the forward channel; 4) After the requested file is downloaded, the user spends some time to digest its content; 5) After the *think* period, the user sends another request to download another file. The ON/OFF cycle keeps on going until the session is finished as shown in figure 5.1. The performance metric directly influencing the user perception is the file response time denoted by $t_{response}$ in figure 5.1, which is the time elapses between the epoch when the request is sent and the epoch when the last packet of the file is received [85].

There are several models proposed in the literature to analyze the ON/OFF interactive data services. The original Engset model is a finite-population, multiple-server model with no buffer space. Heyman [45] adapts the Engset model to analyze the effect of TCP congestion control over an wired access network in which several slow links are multiplexed onto a faster link. Therefore the slow link capacity sets an upper bound of each source's input rate to the fast link. While in our scenario, the bandwidth of the forward and reverse channel is shared among all active users. So it is possible for a user to achieve the full bandwidth if he is the only active user in the network. Schwefel [84] extends Heyman's model [45] into packet-level with a single server and modified ON/OFF arrival process. The performance metrics such as throughput per user, aggregate throughput, queue-length distribution and average number of active users can be computed from Schwefel's model. However this model becomes intractable with many users and power-tailed file sizes due to

the complex and extensive computations it requires [45].

Berger and Kogan [16] develops a closed queuing network model for bandwidth dimensioning for elastic data traffic. The model assumes that under heavy traffic the TCP congestion control can ensure each active user to have a packet in the bottleneck queue. Therefore the utilization of the link is exponentially close to one. By letting the number of users goes to infinity, they obtain close-form dimensioning rules for a single bottleneck link.

In [85], the authors use a simple finite-population analytical model for the shared forward channel, which is applied to web browsing over third generation EDGE (Enhanced Data Rates for GSM Evolution) TDMA system. They modified the classic "machine-repairman" model [91] by adding a delay block which is used to model the overhead delay encountered by the file while being served. Although in a real system the delay overhead could occur at various stages during the file transfer, in this model all the delays are aggregated together into a single delay block for simplicity reasons. The model further assumes that the user performance is constrained only by the forward channel. Therefore bandwidth sharing in the reverse channel has not been addressed.

In this chapter, the transmission time and the propagation delay in both channels are explicitly modeled. Therefore bottlenecks in both channels can be captured. The request delay depends on the reverse channel bandwidth and the propagation delay. For example, one retransmission could increase the request delay by one more round trip time. The file service time in the forward channel depends on the file size and the number of requests currently being served. We adopt a closed queuing model which is very general and can be applied to different wireless access works such as cellular networks [79], wireless metropolitan area networks

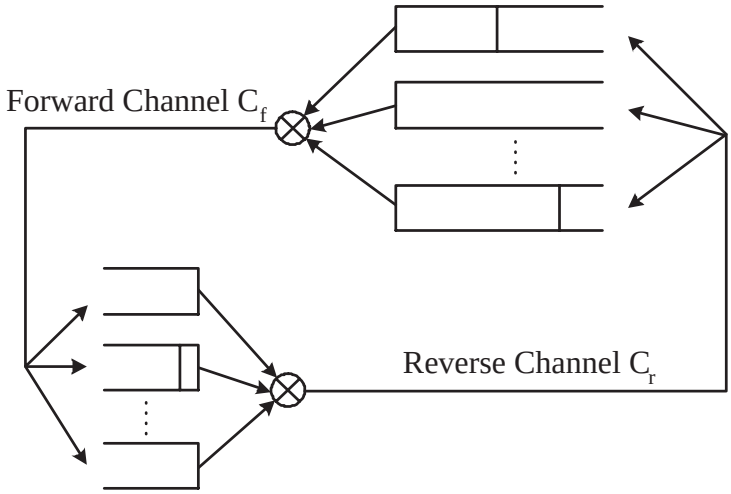


Figure 5.2: Closed queuing network model for interactive data services in wireless access networks

(WMAN) [32] [1] [93] [89] [55] [33] and satellite networks [90]. It can provide insight into the dynamics of the network and into the design of the MAC protocols in the reverse channel. From this model, we can calculate the important performance metrics such as the utilization in the forward channel, the file response time and average user throughput etc.

5.2 System Model

A single user behavior model is shown in figure 5.1 and the system model is shown in figure 5.2. There are N identical ON/OFF users in this closed queuing network. The reverse channel is a multiple access channel and the request delay includes the request transmission time, the reverse channel propagation delay and possibly additional delay caused by retransmissions. Once the request is received, the requested file will be sent over the forward channel. We assume per user queuing is maintained at the hub and a processor sharing discipline such as round robin

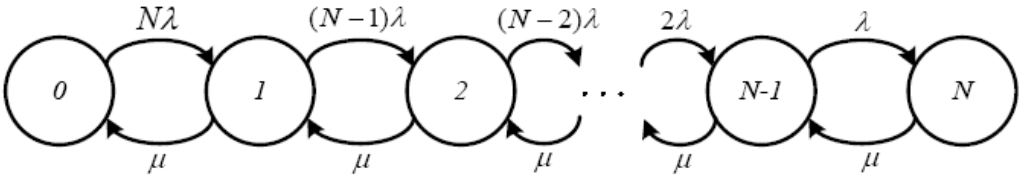


Figure 5.3: The state diagram of the finite-population queuing system

is used in the forward channel. Therefore the service time $t_{service}$ of a requested file depends on its file size, number of files currently being served and the forward channel bandwidth. File sizes are independent and identically distributed with an average length of $E[L_f]$ bits. The propagation delay in the forward channel is t_f. The digest time of a user t_{think} is independent of the file service time $t_{service}$. The bandwidth in the forward and reverse channel is C_f bps and C_r bps respectively.

In our analytical model, we lump the think time, the request delay in the reverse channel and the propagation delay in the forward channel into a single delay called t_{idle} i.e. $t_{idle} = t_{think} + t_r + t_f$ as shown in figure 5.1. It has been shown in [85] [45] that the system performance measures are insensitive to distributions of the file sizes and idle times except through their means. Therefore the results are the same as when the distributions are exponential. In the following we will develop a Markov chain by assuming the idle time and service time are exponentially distributed.

Let us denote by $K(t)$ the number of files currently being received in the forward channel at time t. $K(t)$ is a stochastic process which takes values in the state space $S = \{0, 1, 2, ..., N-1, N\}$. When $K(t)$ equals k where $1 \leq k \leq N$, each of the k files is served at a rate $\frac{C_f}{k}$ bps. Since the files are exponentially distributed with mean file size $E[L_f]$ bits, each file is finished at the rate of $\frac{C_f}{k*E[L_f]}$ [71]. Therefore the rate anyone of them finished is given by $\frac{C_f}{E[L_f]}$.

The Markov chain is shown in figure 5.3 in which:

$$\mu = \frac{C_f}{E[L_f]} \tag{5.1}$$

$$\lambda = \frac{1}{E[t_{idle}]} \tag{5.2}$$

The steady state probabilities of $K(t)$ are given by the following equation:

$$P_k = P_0 \cdot \rho^k \cdot \frac{N!}{(N-k)!} \tag{5.3}$$

where

$$\rho = \frac{\lambda}{\mu} \tag{5.4}$$

$$P_0 = \frac{1}{\sum_{k=0}^{N} \rho^k \cdot \frac{N!}{(N-k)!}} \tag{5.5}$$

The utilization of the forward channel is given by:

$$U = 1 - P_0 \tag{5.6}$$

In the closed queuing network, the total number of users in the system is N, the average delay in the system is $E[t_{response} + t_{think}]$ as shown in figure 5.1 and the average throughput i.e. average number of users entering/leaving the system is $\mu \cdot U$ [85]. By applying Little's law, we have

$$E[t_{response} + t_{think}] = \frac{N}{\mu \cdot U} \tag{5.7}$$

Therefore the average file response time is given by

$$E[t_{response}] = \frac{N}{\mu \cdot U} - E[t_{think}] \tag{5.8}$$

and the average user throughput is defined as follows:

$$E[r] = \frac{E[L_f]}{E[t_{response}]} \tag{5.9}$$

From figure 5.1, we can see that $t_{response} = t_r + t_f + t_{service}$. Therefore $E[r]$ takes into account the reverse channel access delay t_r and the forward channel propagation delay t_f in addition the file service time $t_{service}$. Another important metric is the average service rate denoted by $E[r^{'}]$

$$E[r^{'}] = \frac{E[L_f]}{E[t_{service}]} \tag{5.10}$$

$E[r^{'}]$ gives the average service rate of a file since the user receives its first packet. In the following, we will show that we can calculate the distribution of the service rate which can be used to provide the probabilistic bounds.

From an outside observers' point of view, the probability of k users currently served in the forward channel is give by equation 5.3. However from a user inside the system we need to consider the batch effects [92], the probability of a user being a member of k currently begin served users is given as following

$$P_k^{'} = \frac{k \cdot P_k}{\sum_{k=1}^{N} k \cdot P_k} \tag{5.11}$$

Also note that

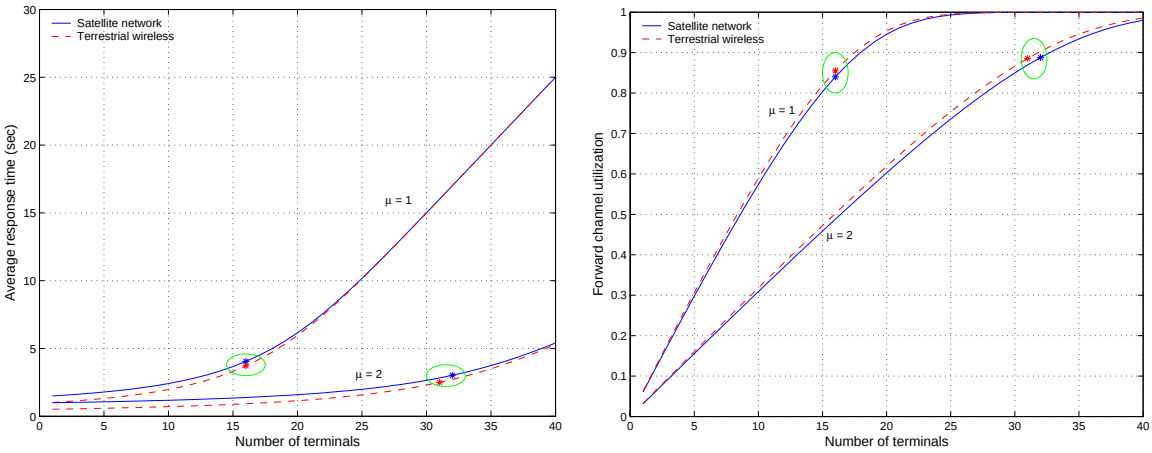
$$P\{r^{'} = \frac{C_f}{k}\} = P_k^{'} \tag{5.12}$$

$$E[r^{'}] = \sum_{k=1}^{N} \frac{C_f}{k} \cdot P\{r^{'} = \frac{C_f}{k}\} \tag{5.13}$$

From the above two equations, the average service rate each user receives and probabilistic estimates about it.

5.3 Analytical and Simulation Results

In the section, we will show how the system performance depends on the forward and reverse channel parameters such as bandwidth, delay and terminal population.



(a) Average file response time

(b) Forward channel utilization

Figure 5.4: Average file response time and forward channel utilization for different terminal populations

We evaluate the performance of several proposed MAC protocols in the reverse channel. After thorough analysis of the forward and reverse channel traffic, a new MAC protocol called CPFDAMA is designed which outperforms existing MAC protocols in term of both channel utilization and file response time.

5.3.1 Bottleneck in the Forward Channel Only

In this section, we release the reverse channel bandwidth constraint so that it only introduces a constant propagation delay. Figure 5.4(a) shows the file response time in terrestrial wireless networks and satellite networks with one way propagation delay of 0.01 ms and 250 ms respectively. The average file size is 64 KB and the forward channel bandwidth is 512 kbps or 1024 kbps correspondingly $\mu = 1$ or $\mu = 2$. The average think time is 15 seconds. These parameters are chosen based on the web traffic models developed in [65] [23] [10] [86] [30].

When there is only one active user in the system, there is no queuing and

the file response time equals the average service time in the forward channel plus the two way propagation delay. As the number of users N keeps on increasing, the forward channel utilization approaches utility as shown in figure 5.4(b). Let's define N^* as following:

$$N^* = 1 + \frac{\mu}{\lambda} \tag{5.14}$$

N^* is called the *saturation number* [91]. As long as the terminal population N is less than N^*, the response time only increases gradually with N due to the statistical multiplexing gain. However when N is greater than N^*, the system soon becomes overloaded and the response time increases almost linearly with N. With the same file service rate μ in the forward channel, the response time and utilization are very similar to each other in terrestrial wireless and satellite networks. When the service rate is doubled, the saturation number N^* is also doubled and the response time becomes smaller when the terminal population is less than N^*.

Figure 5.5(a) shows the average throughput in the forward channel for different terminal populations. It shows that the average throughput is much smaller in satellite networks than in terrestrial wireless networks when the number of terminals is small. The is because the two way propagation delay becomes a significant component in the total response time when the service time is small. After the terminal population is increased above N^*, the service time becomes much larger than the propagation delay therefore both of them have similar throughput.

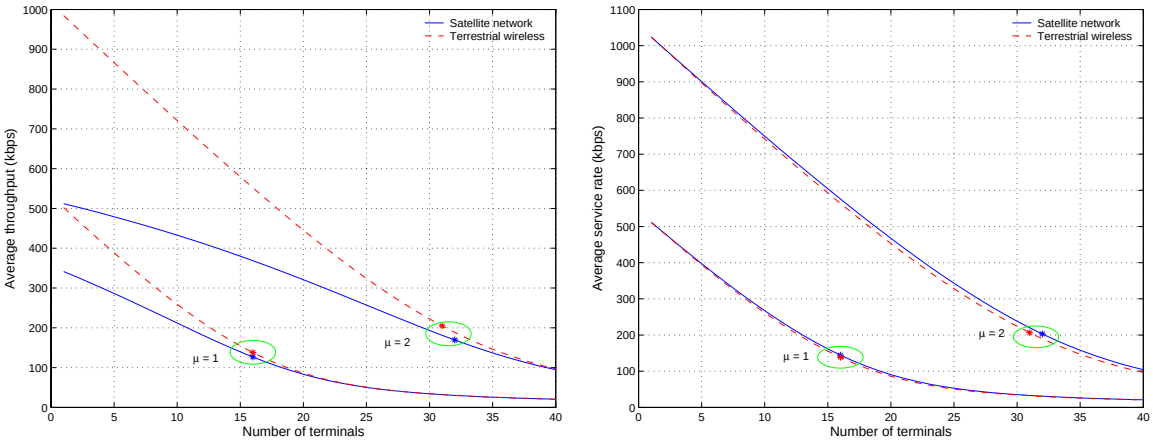
The average service rate shown in 5.5(b) is the packet arrival rate since the user receives the first packet in a file. This figure shows that the average service rates in both networks are very close to each other. This figure can be used to dimension the networks. For example, in order to provide average service rate of

no less than 400 kbps, forward channel bandwidth of 512 kbps i.e. $\mu = 1$ can serve up to 5 terminals. When the forward channel bandwidth is increased to 1024 kbps i.e. $\mu = 2$, it can provide the same average service rate to 22 terminals. On the other hand, if the terminal population and the average service rate requirement are known, it is also easy to find out how much bandwidth is needed. For example, if there are 15 terminals and the mean service rate of each should be greater than 600 kps, from figure 5.5(b) we can find out that the forward channel bandwidth should be at least 1024 kbps.

In addition to the average service rate, we can get the rate distributions for different terminal populations through equation 5.12. Figure 5.6(a) and figure 5.6(b) show the service rate distributions for different terminal populations which can be used to provide probability based service. For example, figure 5.5(b) shows that the average service rate is around 600 kbps when the number of terminals is 15 and the forward channel bandwidth is 1024 kbps. The lower plot in 5.6(a) shows that with probability more than 30% each terminal will receive a service rate of 1024 kbps while with probability of less than 10% the service rate will be less than 256 kbps.

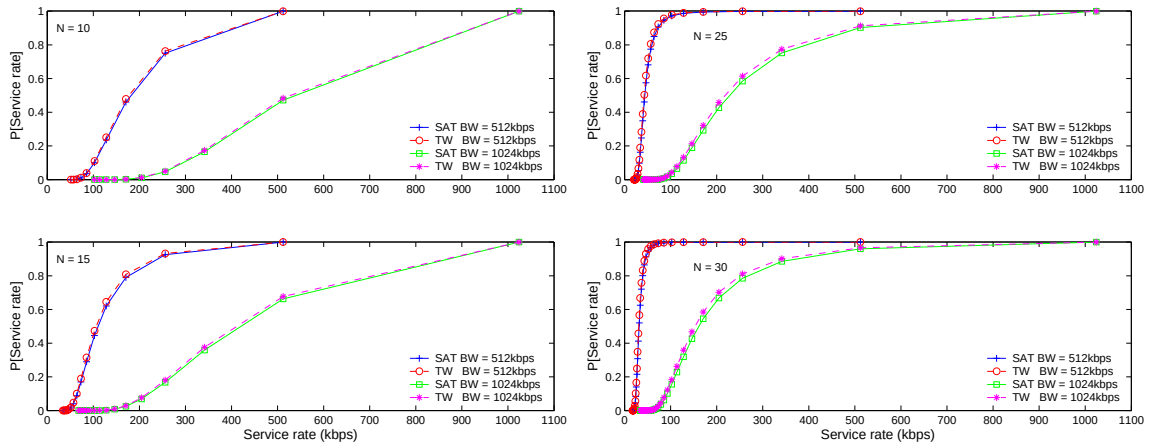
5.3.2 MAC Protocols in the Reverse Channel

In this section, we will describe four MAC protocols in the reverse channel which include static TDMA, slotted Aloha and two reservation based protocols Aloha/periodic stream and CFDAMA. Advantages and disadvantages are also pointed out for each protocol in term of efficiency and delay for bursty traffic.



(a) Forward channel average throughput (b) Forward channel average service rate

Figure 5.5: Average throughput and service rate in the forward channel for different terminal populations



(a) N = 10 and 15 (b) N = 25 and 30

Figure 5.6: Forward channel service rate distributions for different terminal populations

Static TDMA

In static TDMA when there are N terminals in the network, each terminal is assigned a time slot every N slots for the life time of the terminal [47]. Therefore the slots assigned to a terminal will be wasted if it has no packets to send.

Slotted Aloha

In slotted Aloha, all the terminals are synchronized with each other and MAC packets are sent only at the beginning of a slot. If a MAC packet is received successfully by the hub, an acknowledgment will be sent back to the terminal and the packet is purged from the retransmission buffer. However if a collision happens, the terminal will timeout after one round trip time and retransmit the packet after a random backoff time. An exponential backoff strategy is used here.

In slotted Aloha, the average number of attempts for a successful transmission is e^G, where G is the average load in reverse channel including both new arrivals and retransmissions. The aggregate arrivals is assumed to be a Poisson process. The average access delay of the slotted Aloha denoted by D_{SA} [54] is

$$D_{SA} = 1 + t_{prop} + (e^G - 1)(1 + 2 * t_{prop} + t_{bkoff}) \tag{5.15}$$

Where t_{prop} is the normalized propagation delay and t_{bkoff} is the normalized backoff time. Both are normalized to the packet transmission delay. If the offered load G is heavy, collisions will increase the packet delay dramatically as shown in equation 5.15. Therefore to deliver the packets with short delay, the slotted Aloha channel bandwidth should be sized such that it operates in the low load region most of the time.

Aloha/periodic Stream

Aloha/periodic stream [100] is a reservation based protocol. After new packets arrive at the empty queue of a terminal, the terminal becomes active and an Aloha request will be sent to the hub. After the Aloha request is received, the terminal is assigned periodic bandwidth. If the persistent backlog packets exceed some threshold during the active period, additional bandwidth is requested by piggybacking the request in the data packets. Additional bandwidth is provided until the maximum is attained or the backlog is decreasing. If the terminal hasn't transmitted traffic for a period of time, the terminal will be inactive. The bandwidth is given an inactivity timeout value. If no packet arrives from the terminal during the timeout period, the bandwidth assigned to it will be released.

The Aloha/periodic stream scheme tries to explore the regeneration cycles of the reverse channel traffic. However due to the bursty nature of the traffic, the assigned channel bandwidth to the terminal is wasted if there are no packets arriving at the terminal during the timeout period. The timeout parameter plays an important role in this protocol in term of efficiency and delay. If the timeout is set relatively long, the wasted bandwidth could increase, especially for bursty Internet traffic as in the case we are interested in. On the other hand, if the timeout value is set too small, it will increase the frequency of the request and release cycles. This will increase the packet delay due to the request phase overhead. At the same time more frequent requests will increase the Aloha request channel traffic load and lead to more collisions which eventually will increase access delay dramatically especially in networks with large propagation delay e.g. satellite networks. Besides the timeout setting problem, the period during which the channel release message propagates to a specific terminal is still hold by the terminal and cannot be used

by other terminals. This problem becomes more significant with the increase of the propagation delay.

CFDAMA

In combined free demand assignment multiple access (CFDAMA) [60], the reverse channel is simply divided into equal-size time slots; each time slot can accommodate one MAC packet. A terminal sends its data packet in its assigned time slot. A MAC packet has two parts: header and payload. The header contains addressing and routing information as well as the bandwidth request. There are three possible request schemes. A terminal can send its bandwidth request in a preassigned, or random access request slot or piggyback in its data packet. Preassigned scheme requires each terminal has a dedicated request channel. Random request scheme allows all the terminals to access the same request channel with the possibility of collisions. The authors argue that piggybacking the request in the data packet header is the most efficient request strategy.

The bandwidth scheduler is located at the hub on the ground. Therefore the request delay is 250 ms. The bandwidth scheduler maintains a reservation table and a free assignment table. The reservation table contains the number of requested slots by different terminals. Each time a request is received, the scheduler will place an entry in the bottom of the reservation request table. The free assignment table contains the IDs of all terminals in the network. At first, the scheduler will serve the reservation request table by assigning contiguous slots to the corresponding terminals based on the number of slots requested [72]. When there is no demand, the scheduler allocates the remaining free bandwidth to the terminals according to some scheduling schemes such as round robin. In order to provide those terminals

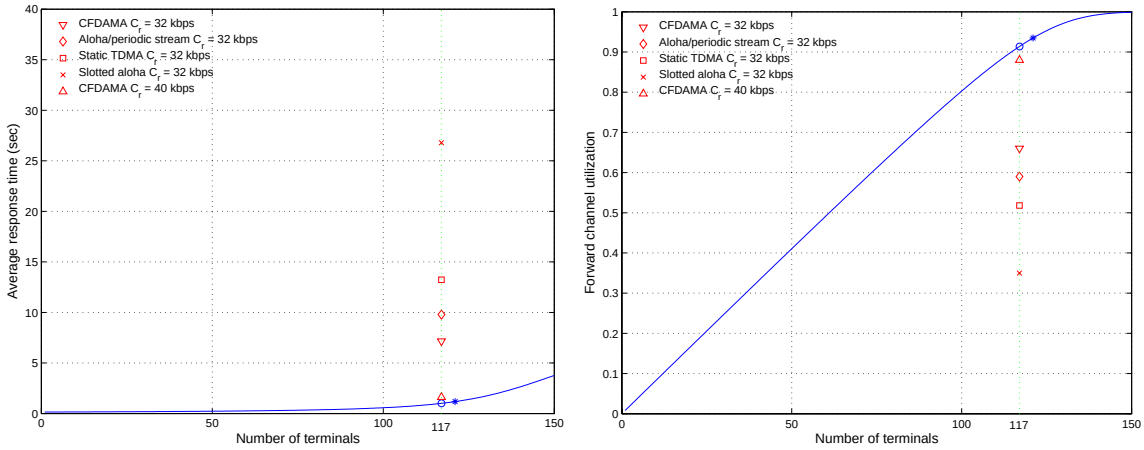
that have been idle for a long time a higher probability to obtain a free assigned slot, each time the scheduler assigns slots to a terminal, the corresponding entry in the reservation table is removed and the corresponding terminal ID is also moved to the bottom in the free table.

When the channel is lightly loaded, the probability of a terminal obtaining free assignment is high therefore small packet delay can be achieved. However the probability of receiving free bandwidth depends on the population of the terminals and the delay performance degrades with the increase of the population size. When the reverse channel is heavily loaded, CFDAMA behaves like a reservation scheme and high channel efficiency can be achieved. In [72], CFDAMA has been evaluated with three different traffic models. It has been shown that the utilization and delay performance for Poisson traffic is almost the same for different request schemes. The performance of CFDAMA with Pareto ON-OFF and exponential ON-OFF traffic source is very similar. This gives the evidence that the ON-OFF nature of a traffic model plays a more significant role in MAC protocol evaluation than the specific distribution of ON and OFF times.

5.3.3 Bandwidth Requirement for Request Traffic in the Reverse Channel

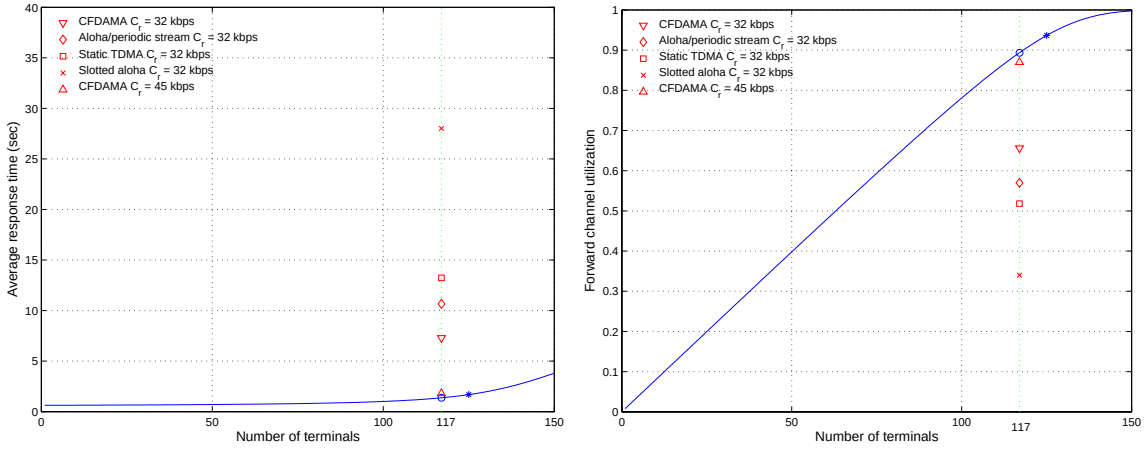
In the stable state the file service rate in the forward channel equals the request service rate in the reverse channel. Assume perfect scheduling in the reverse channel i.e. the hub knows the distributed queue status at the terminals without delay, we have the following equation

$$\frac{C_f}{E[L_f]} = \frac{C_r}{E[L_r]} \tag{5.16}$$



(a) File response time (b) Forward channel utilization

Figure 5.7: The effects of bottleneck in the reverse channel (terrestrial wireless networks)



(a) File response time (b) Forward channel utilization

Figure 5.8: The effects of bottleneck in the reverse channel (satellite networks)

In which C_f and C_r are the forward and reverse channel bandwidth; $E[L_f]$ and $E[L_r]$ are the average file and request sizes. We assume the requests have a fixed size of 432 bytes [65]. In section 5.3.1, we evaluate the system performance of a 4 Mbps forward channel without reverse channel bandwidth constraint. From equation 5.16, we can get the reverse channel bandwidth should be at least 27 kbps in order not to become the system bottleneck.

The file response time and the forward channel utilization are shown in figure 5.7(a) and figure 5.7(b) for four different MAC protocols. With the reverse channel bandwidth of 32 kbps in terrestrial wireless networks, CFDAMA performs the best which can achieve a forward channel utilization of 66% and file response time of 7.2 seconds. As expected Aloha/periodic stream performs better than static TDMA due to the dynamic release of slots hold by idle users after the timeout. Because the bandwidth is so limited, a lot of collisions occur in slotted Aloha which causes the response time increased dramatically. With 32 kbps bandwidth, the reverse channel is the system bottleneck for the above four MAC protocols. We increase the reverse channel bandwidth to 40 kbps in terrestrial wireless networks and to 45 kbps in satellite networks so that CFDAMA can achieve a utilization close to 90%, which corresponds to the utilization without reverse channel constraint. Figure 5.8(a) and figure 5.8(b) show similar results in satellite networks. The performance of Aloha/periodic stream and CFDAMA is worse in satellite networks compared with terrestrial wireless networks because the queue status in each terminal has been delayed longer.

Table 5.1 shows the forward channel utilization and file response time for different channel capacity and different number of terminals . This table can be used by the network designer for capacity planning. For example, assume the forward

channel bandwidth is 4096 kbps and combined free DAMA is used in the reverse channel, if the users require a file response time less than 7.29 seconds, the reverse channel bandwidth should be at least 32 kbps. Under such situation, the reverse channel is the system bottleneck. Therefore when its bandwidth is increased to 45 kbps, the response time drops to 1.81 seconds.

5.3.4 Effects of Acknowledgement Traffic in the Reverse Channel

In the previous section, we only consider the request traffic in the reverse channel. For web browsing, the traffic in the reverse channel also includes the transport layer acknowledgements as shown in figure 5.10. Assuming perfect scheduling and following the same argument in the previous section, the ACK rate should equal the data rate to achieve high utilization in the forward channel

$$\frac{C_f}{2 \cdot L_{data}} = \frac{C_r}{L_{ack}} \tag{5.17}$$

in which L_{data} is the forward channel data packet size and L_{ack} is the reverse channel acknowledgement size. The terminal sends an ACK every two data packets received as in TCP. From equation 5.17, we can calculate the bandwidth required in the reverse channel for a given C_f. For example, if L_{ack} = 40 bytes and L_{data} = 552 bytes which include 40 bytes TCP/IP header and 512 bytes payload. For forward channel bandwidth of 4 Mbps, the bandwidth required for ACKs in the reverse channel is at least 148.4 kbps.

It should be pointed out that there exists a close correlation between forward channel data packets and the reverse channel acknowledgement packets, i.e. the acknowledgement packets are triggered by the data packets. In TCP, an acknowledgement is sent every two data packets are received. Regardless how bursty and

Table 5.1: The forward channel utilization and file response time for different channel capacity and different number of terminals. SA: Slotted Aloha, ST: Static TDMA, AP: Aloha/periodic stream, CF: Combined free DAMA

N	C_f (kbps)	C_r (kbps)	MAC	Utilization	Response time (sec)
10	512	10	CF	57.42%	2.42
20	512	14	CF	94.54%	6.16
40	512	30	CF	100%	25
10	1024	14	CF	30.90%	1.18
20	1024	30	CF	60.26%	1.60
30	1024	50	CF	84.99%	2.65
40	1024	80	CF	98.00%	5.41
40	4096	80	CF	31.89%	0.6805
80	4096	80	CF	63.20%	0.8227
117	4096	32	SA	34%	28.01
117	4096	32	ST	51.8%	13.23
117	4096	32	AP	57%	10.66
117	4096	32	CF	65.6%	7.29
117	4096	45	CF	87%	1.81

Table 5.2: Reverse channel packet types and proposed MAC schemes for web browsing

Packet Type	Proposed MAC scheme
Transport Layer ACKs	Polling
HTTP Requests	CFDAMA

unpredictable of the forward channel traffic is, this behavior will always hold and be predictable [26]. This motivates us to design a new MAC protocol CPFDAMA which explores this characteristics as described in the following.

5.3.5 CPFDAMA

We classify the reverse channel packets into two categories as in table 5.2 by the transport layer port numbers and the packet sizes, and each type of packets is served by a different MAC protocol.

The system model of our protocol called combined polling, free demand assignment multiple access protocol (CPFDAMA) is shown in figure 5.9. A message coming from the upper layer is first segmented and then buffered in one of the two MAC layer queues. The reverse channel bandwidth controller is located at the hub and it assigns reverse channel bandwidth to the terminals based on the requests they've made.

In CPFDAMA, the reverse channel bandwidth is divided into two parts. One part is used to transfer HTTP requests and CFDAMA is used as the MAC protocols. Another part of the bandwidth is used for the transport layer acknowledgements and polling is used for them. The boundary between them is fixed. For

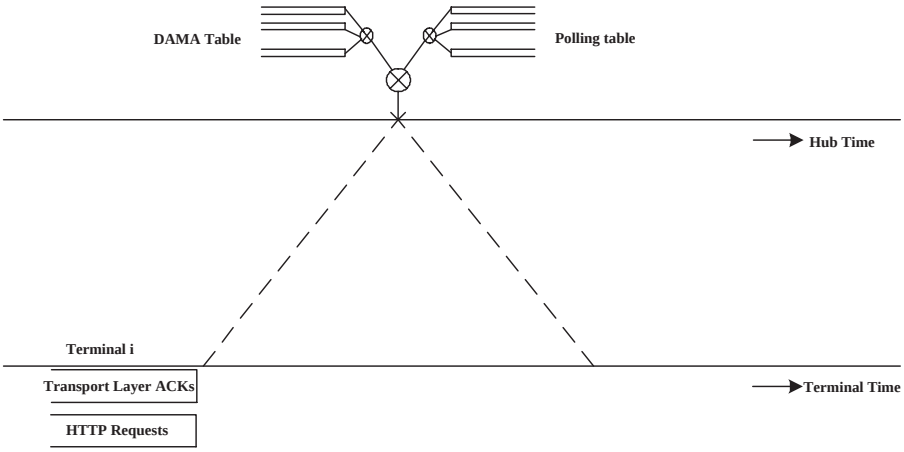


Figure 5.9: System model of combined polling, free and demand assignment multiple access (CPFDAMA) protocol for web browsing

example in every five slots, the first to the fourth slots are assigned to ACK traffic and the last slot is assigned to HTTP request traffic.

In the transport layer, the sender polls the receiver to send an acknowledgement by setting a bit in the packet header. This one bit information serves as a reservation request for the ACK in the reverse channel. The MAC layer at the hub will check the packet header of every packet. If the polling bit is set, an entry will be added in the polling table at hub controller. It will try to assign enough slots for transmitting the transport layer acknowledgements. Therefore after a terminal receives a polling data packet, it will receive enough slot assignments to enable it to transmit the acknowledgements. The idea for this scheme is to try to change MAC problem into classical centralized scheduling problem by prediction. The advantages of this approach compared with CFDAMA are as following. The ACK traffic does not need to send bandwidth request in the reverse channel which decreases the bandwidth request delay and at the same time reduces the control channel traffic. Therefore less control channel bandwidth is required. Because

there is no bandwidth request overhead, the delay performance of ACK traffic is also improved.

For the HTTP request traffic, it will behave as in CFDAMA. A terminal sends bandwidth requests to the hub and the hub assigns bandwidth to each terminal according to some scheduling scheme. The hub bandwidth controller has two tables. One is for the CFDAMA and the other is for the polling. The CFDAMA table contains the requesting terminal ID numbers and their corresponding requested slots. The polling table contains terminal ID numbers and the number of slots needed to send the ACKs. Whenever a table becomes empty, the hub can assign free bandwidth to terminals in a round robin manner. Following the example in the previous paragraph, let's assume one frame has five slots. The first four assigned to ACK and the last one assigned to HTTP requests. If the CFDAMA tables becomes empty, the controller can assign the last slot in the current frame as a free slot to a terminal and the same slot in the next frame to another terminal and so on. This process continues until there is a new entry added to the CFDAMA table. If a terminal receives a free slot while the HTTP request queue is empty, it can send ACK in that slot to improve the utilization and delay performance. The same approach applies to the polling table.

In section 5.3.3 and section 5.3.4, 40 kbps is need for HTTP requests in CFDAMA to achieve a utilization close to 90% in terrestrial wireless networks. The bandwidth needed for ACK traffic is at least 148.4 kbps. Since the slot size is 48 bytes and the size of an acknowledgement is 40 bytes, 8 bytes in a slot has been used for padding. Therefore the bandwidth requirement has to be increased when taking the padding into account. The bandwidth required by the ACK traffic now becomes 48/40*148.4 kbps i.e. 178.08 kbps. In the following simulation, we

choose 40 kbps for the HTTP request bandwidth and 200 kbps for ACK bandwidth. Therefore the total bandwidth in the reverse channel is 240 kbps. Each reverse channel frame contains 6 slots. One slot is for HTTP request traffic and the other five used for ACK traffic. As in the terrestrial wireless networks, we can get the bandwidth requirement for satellite networks. The satellite networks use a 45 kbps for HTTP request bandwidth and 180 kbps for ACK bandwidth. Each reverse channel frame contains 5 slots. One slot is for HTTP request traffic and the other four used for ACK traffic.

Table 5.3 shows the file response time and forward channel utilization in terrestrial wireless networks considering both the request and acknowledgement traffic. Because the acknowledgement traffic increases the queuing delay of the requests, the file response time increases correspondingly compared with figure 5.7(a). At the same time, the forward channel utilization has decreased due to the smaller request throughput in the reverse channel. From this table, we can see that CPFDAMA can achieve the best performance in both channel utilization and file response time. It should be noted that it also outperforms all the other protocols in term of ACK throughput and ACK delay. Similar results are shown in table 5.4 for satellite networks. Therefore CPFDAMA is an efficient MAC protocol for web browsing in access networks.

5.4 Web Browsing in Wireless Access Networks: A System Approach

In the above sections, we implicitly assume that the files are located at the hub which is true if the file servers are located in the same LAN with the hub in which

Table 5.3: The effect of acknowledgements in the reverse channel (terrestrial wireless networks)

MAC protocols	Utilization (percentage)	Response time (sec)	ACK throughput (kbps)	ACK delay (sec)
Slotted Aloha	33%	27.6	76	12.3
Static TDMA	70.1%	4.98	125	5.36
Aloha Periodic	78.0%	4.2	146	4.0
CFDAMA	82.2%	2.18	150	1.6
CPFDAMA	90.0%	1.40	166	0.15

Table 5.4: The effect of acknowledgements in the reverse channel (satellite networks)

MAC protocols	Utilization (percentage)	Response time (sec)	ACK throughput (kbps)	ACK delay (sec)
Slotted Aloha	32.1%	29.8	70	13.7
Static TDMA	70.2%	5.44	125	5.6
Aloha Periodic	76.8%	4.4	141	4.3
CFDAMA	81.3%	3.2	150	2.3
CPFDAMA	88.0%	1.60	155	0.45

the file transfer delay between the file servers and the hub is negligible. While for web browsing, the hub are connected to the web servers through a wide area network and the delay between them has to be taken into account to get accurate performance results. In this section, we will consider the web browsing in wireless networks from an end-to-end perspective.

To improve web browsing performance in wireless networks, three layers need to be considered. First, at the application layer, a web traffic model is needed to generate realistic source traffic; Second, a transport layer protocol is needed which can solve the problems of TCP over wireless links; Third, a MAC protocol is needed to work efficiently with the given source traffic characteristics and the transport layer. The contribution of our approach is that we are the first to address this problem in a systematic manner rather than focus on a specific layer. Our goal is to analyze web browsing performance in wireless access networks and to improve its performance by designing appropriate protocols.

5.4.1 Web Traffic Modeling

For web browsing, the main traffic loads in the wireless networks are flowing from the Internet to the end users in the forward channel. The traffic flowing in the reverse channel mainly composes some control packets and small data packets [26] [65] such as TCP acknowledgement packets and HTTP request packets as shown in figure 5.10.

A good web traffic model is essential for simulations and experiments to investigate end-to-end performance such as page response time. Recent studies [28] [27] show that web traffic shows self similarity, which means that the traffic is bursty over wide range of time scales and therefore has long range dependence in con-

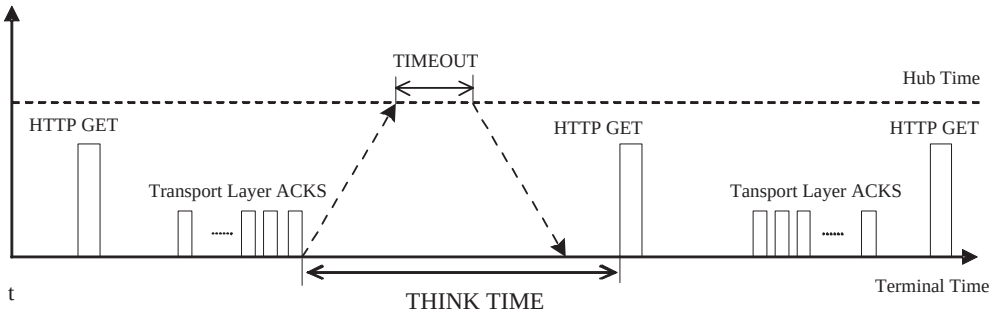


Figure 5.10: Reverse channel traffic characteristics for web browsing

trast to traditional Poisson traffic models which are short range dependent. Self similar traffic can be modeled as superposition of many independent and identically distributed ON/OFF sources whose ON and OFF times have a heavy tailed distributions. Crovella [28] shows evidence that a number of file distributions on the web exhibit heavy tail distributions, including files requested by users, files transmitted through the network and files stored on the servers.

A random variable X follows a heavy tailed distribution if

$$P[X > x] \sim x^{-\alpha} \;\; as\; x \to \infty,\; 0 < \alpha < 2,$$

That is, regardless of the behavior of the distribution for small values of the random variable, if the asymptotic shape of the distribution is hyperbolic, it is heavy tailed. The simplest heavy tailed distribution is the Pareto distribution, with probability density function

$$p(x) = \alpha k^{\alpha} x^{-\alpha-1} \;\; where\; \alpha, k > 0, x \geq k$$

and cumulative distribution function

$$F(x) = P[X \leq x] = 1 - (k/x)^{\alpha}$$

The parameter k is the location parameter and it represents the smallest possible value of random variable X. For Pareto distribution, if $\alpha \leq 2$, it has infinite

variance; if $\alpha \leq 1$, it has infinite mean. While $1 < \alpha < 2$, the mean of Pareto distribution is $\alpha/(\alpha - 1) * k$ which is of interest to us.

HTTP is a request-response based protocol. There are several empirical web traffic models proposed in the literature [65] [23] [10] [86] [30], these models are based on the traffic traces collected either in a local area network or in a wide area network. The elements of an HTTP model are: 1) HTTP request length; 2) HTTP reply length; 3) user think time between retrievals of two successive pages. Mah and Smith [65] [86] argue that it is not sufficient to simply transmit data into the network according to these traffic models. This is because these application-dependent but network-independent web traffic models should be layered over TCP so that the sizes and timing of the packets can be modeled accurately. In our web traffic model, a user begins his web browsing session by sending an HTTP request to the web server. The request length will be fixed of 432 bytes. After the server receives the request, it replies with a page which size will be sampled from a Pareto distribution with with $\alpha = 1.01587$ and $k = 4$ KB. The user think time will be modeled also by Pareto distribution [10] with $k = 5$ sec and $\alpha = 1.5$.

5.4.2 Improve Web Performance by designing a New Transport Layer Protocol

Considering the interoperability issue, we adopts the connection splitting based scheme [90] [11] [44] which is currently used in the industry, and design a new transport layer protocol for reliable data transfer over the wireless link.

Proxies are put at the boundary of the wireless and wired networks. An end-to-end TCP connection is split into three connections at the proxies. The first one is set up between the server and the upstream proxy; the second is from upstream

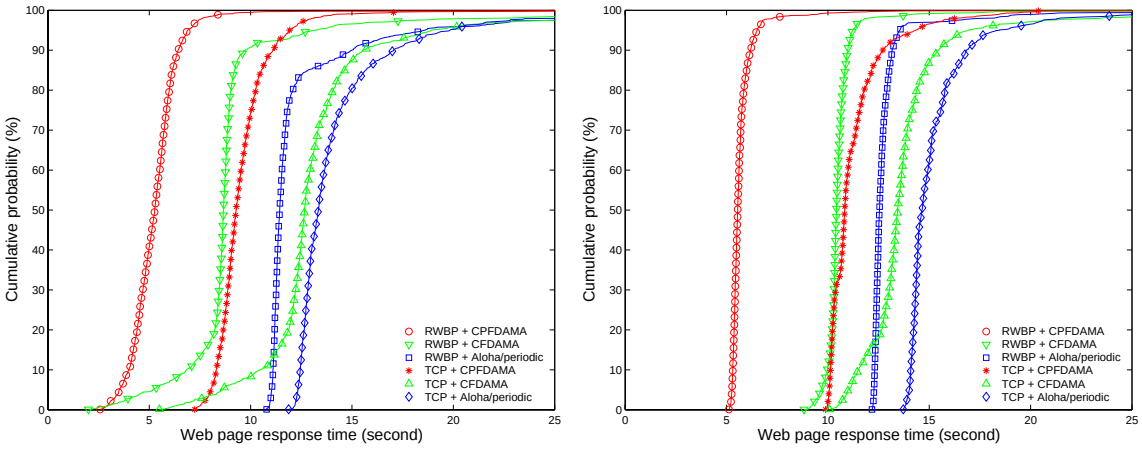
proxy to downstream proxy; and the third is from the downstream proxy to the client. Normal TCP is used for the server-proxy and proxy-client connections. Receiver Window Backpressure Protocol (RWBP) is designed for the proxy-proxy connection to transfer data over the wireless channel.

The upstream proxy and the hub are connected with a fast link e.g. token ring and the propagation delay between them is negligible. A single FIFO queue is maintained for TCP traffic at the hub. While the proxies keep a per-flow queue and they have to buffer the packets waiting for transmission as well as those packets that have been transmitted but not acknowledged. TCP uses slow start to probe the bandwidth at the beginning of a connection and uses additive increase and multiplicative decrease (AIMD) congestion avoidance to converge to fairness in a distributed manner. RWBP is based on TCP; however RWBP cancels all the congestion control algorithms in TCP and uses per-flow queuing, round robin scheduling [22] and receiver window backpressure for congestion management.

Flow control is done between the downstream proxy and the upstream proxy at the transport layer by using the receiver window. For each RWBP connection, the downstream proxy advertises a receiver window based on the available buffer space for that connection just as in TCP. Similarly flow control is also done between the hub and the upstream proxy at the link layer. The hub advertises a receiver window to the upstream proxy so that its queue will not overflow. Using backpressure, RWBP eliminates congestion losses inside the wireless networks.

5.4.3 Performance Evaluations of Web Browsing

In this section, we evaluate the performance of web browsing in a wireless access networks with OPNET. The metric we are interested in is web page response time.



(a) Terrestrial wireless networks

(b) Satellite networks

Figure 5.11: Web page response time for different transport layer and MAC layer protocols.

There are 117 clients in the network to download web files from 117 Internet servers. The forward channel bandwidth is 4 Mbps and the reverse channel bandwidth is 240 kbps in wireless networks and 235 kbps in satellite networks. The link bandwidth from each server to the upstream proxy is 4 Mbps; and the link bandwidth from each downstream proxy to its corresponding client is also 4 Mbps. The link delay from each server to the upstream proxy is 40 ms and the link delay from each downstream proxy to its corresponding client is 0.01 ms. The one way propagation delay between the terminals and the hub is 0.01 ms in terrestrial wireless networks and 250 ms in satellite networks. The gateway buffer size is set to the wireless bandwidth delay product [99]. The maximum segment size is 512 bytes.

We evaluate the web access performance for different transport layer protocols and MAC layer protocols by using the realistic web user behavior model described in section 5.4.1. The results for terrestrial wireless and satellite networks are shown in figure 5.11(a) and figure 5.11(b) respectively. For the transport layer protocols,

RWBP always outperforms TCP when used for the wireless connections because RWBP doesn't need to go through the slow start phase and it eliminates congestion losses inside the wireless networks. For the MAC layer protocols, Aloha/periodic stream gives the worse performance therefore it is not suitable for bursty reverse channel traffic. From figure 5.10 we can see, during the timeout period if there are no packets arriving at a terminal, the assigned channel bandwidth to the terminal is wasted. Actually the period during which the channel release message propagates from the hub to the terminal cannot be used by any terminal either. While in CFDAMA, the hub can assign the free bandwidth to those terminals with more traffic to send. Therefore CFDAMA can achieve higher efficiency and smaller delay than Aloha/periodic stream. However in CFDAMA, the HTTP requests share the bandwidth with the acknowledgements which could increase their delay. Therefore the throughput of the HTTP requests in the reverse channel decreases correspondingly. While in CPFDAMA, HTTP requests and ACKs are served with different MAC protocols. The ACK traffic causes no interference on the HTTP request traffic. Figure 5.11(a) and figure 5.11(b) show RWBP combined with CPFDAMA achieves the best web browsing performance in both short and long delay networks.

In the above, we evaluate web performance over a multiple access wireless channel by using a realistic web traffic model. For the MAC layer protocols, CPFDAMA performs better than Aloha/periodic stream and CFDAMA. For the transport layer, we adopt a new protocol called RWBP which does not have slow start and eliminates congestion losses inside wireless networks. We compare its performance with TCP over the three multiple access protocols. Our results show that RWBP always performs better than TCP in term of web page response time.

Table 5.5: Reverse Channel Traffic for Each Web Page with Different Acknowledgement Frequency

N	Num of ACKs	Num of HTTP GETS	Total Traffic (bytes)	Normalized Total Traffic
2	64	1	64*40 + 432 = 2992	100%
4	32	1	32*40 + 432 = 1712	57.2%
8	16	1	16*40 + 432 = 1072	35.8%

RWBP combined with CPFDAMA achieve the best performance among all the combinations.

5.5 Discussions

5.5.1 Reducing Reverse Channel Traffic Load

Because RWBP does not use acknowledgements to clock out data packets like in TCP, less frequent acknowledgements are needed in the reverse channel. In RWBP, an acknowledgement is sent when every N data packets are received. By increasing N, we can decrease the acknowledgement frequency. According to the web traffic model in section 5.4.1, the average page size is 64 KB. Because the data packet size is 512 bytes, on the average each page contains 128 data packets and $128/N$ acknowledgements are needed for each page. One HTTP Get is needed for each page. The size of HTTP Get packets is 432 bytes and the size of each acknowledgement packet is 40 bytes. Table 5.5 shows the reverse channel traffic load for different acknowledgement frequency. In [99], we have shown that N

can be increased up to eight without degrading the forward channel performance. Only after N is increased beyond sixteen, we see that the throughput in the forward channel begins to decrease. By increasing N from two to eight, the reverse channel traffic can be reduced by about 64%. When the acknowledgement traffic load is reduced, smaller delay can be achieved for both ACKs and HTTP Gets which leads to improved response time performance.

5.5.2 Extensions to Support More Applications

In the previous sections, we focus on only one of the Internet applications, specifically web browsing. As shown in table 5.6, our scheme can be extended to support most of the popular Internet applications. Besides web browsing, sometimes a user may produce and transmit packets in a sporadic bursty manner as in Telnet, point of sale transaction. This kind of interactive applications is more sensitive to packet delay. Depending on the propagation delay, the network designer can choose to use different protocols. In terrestrial wireless networks, the terminals can use volume based reservation. Since the propagation delay is small, they can be delivered in a timely manner as long as the requests are scheduled with high priority. However since the propagation delay is long in satellite networks, the reservation overhead becomes too expensive. Random access can be used without violating the delay constraint as long as random access channel bandwidth is sized appropriately. It should be noted that random access is still an option to deliver short messages in terrestrial wireless networks.

For an FTP download, it is very similar to a web transfer with a relatively large file size. Occasionally a user may upload a large file using FTP or send an email with a large attachment using SMTP. This kind of applications will generate bulk

Table 5.6: Reverse channel packet types, QoS and proposed MAC schemes for Internet applications

Packet Type	QoS	Proposed MAC scheme
Short HTTP GET, Telnet POS, DNS lookup	Delay sensitive	Random access or Reservation (high priority)
Bulk data packets (FTP upload, SMTP)	Delay insensitive throughput sensitive	Reservation
Bulk ACK packets of HTTP, FTP downloads	Both delay and throughput sensitive	Polling

data packets for the reverse channel and they are not sensitive to the packet delay. Throughput is more important to them. The long HTTP requests also fall into this category. For the last category of the reverse channel traffic i.e. the transport layer acknowledgements, polling can be used as described in CPDAMA.

5.5.3 Congestion and Flow Control

In the following, we will show how the MAC protocol proposed above can be integrated with the congestion control in the transport layer to form an end-to-end resource management mechanism. The hub can keep on monitoring the reverse channel status. If congestion is detected in the random access channel, the hub notifies the terminals about the congestion status and the short messages will be enqueued in the reservation queue with some probability depends on the load. This could release the temporary congestion in the random access channel.

If there are a large amount traffic coming from bulk data transfer, the reverse

channel could be congested and the reservation queue size will increase. In oder not to overload the reverse channel and cause packet losses, the MAC layer monitors the average queue size with a low pass filter. It will send a congestion notification to the upper layer once the reservation queue size increases above a higher threshold. After the upper layer receives the congestion notification, it should reduce its sending rate. After the congestion is released and the queue size decreases below a lower threshold, the MAC layer can notify the upper layer to speed up. This cross layer design method can be integrated with the transport layer protocol to form a systematic traffic management mechanism.

5.6 Summary

In this chapter, we study the capacity planning and protocols design in wireless access networks for web browsing. We've developed a closed queuing model which can capture the bottleneck effects in both forward and reverse channels. Based on the model, we calculate the number of users that can be supported for a given forward channel bandwidth without reverse channel constraint. We then evaluate the system performance of four MAC protocols using realistic web traffic model. In order to improve the web browsing performance, we developed a new MAC protocol and a new transport layer protocol. Our MAC protocol called CPF-DAMA explores the correlation between the forward channel data packets and the reverse channel acknowledgement traffic. Our transport layer protocol RWBP uses per-flow queuing, round robin scheduling and receiver window backpressure for congestion management. We have shown through analysis and simulation that CPFDAMA is a very efficient MAC protocol for web browsing in wireless access network. It can maintain high utilization in the forward channel and can deliver

the HTTP requests and acknowledgements with higher throughput and smaller delay compared with existing MAC protocol. The results we get in this chapter can be use for the network designer to dimension their networks and to provide QoS to certain number of users.

In this chapter, we assume all the users are identical. In the future, we would like to investigate users with different channel conditions. We would like to extend our scheme into multiple classes. It is also interesting to combine our scheme with physical layer scheme such as adaptive modulation and adaptive coding [53] together to further improve the system performance.

Chapter 6

Conclusions

In this chapter, we would conclude this dissertation and summarize our contributions. We've addressed three problems in our work.

Because it is difficult for an end-to-end scheme to solve the reliable communication problems in the satellite networks, we propose a connection splitting based solution. An end-to-end connection is split into three connections. A reliable protocol RWBP is designed for the satellite connection by taking advantage of the specific characteristics of the satellite network. RWBP uses per-flow queuing, round robin scheduling and receiver window backpressure for congestion control. The newly designed error control scheme in RWBP can effectively recover not only the first time losses but also the higher order losses. RWBP requires less reverse channel bandwidth without sacrificing the forward channel throughput. RWBP can still maintain good performance even when there is high priority traffic such as streaming video, audio competing with the TCP traffic in the satellite networks. It is seamlessly integrated with terrestrial TCP and the congestion inside the satellite network can be fed back to the Internet hosts.

We have developed a multichannel random access protocol (FMCSA) for short

messages with delay constraints in the reverse channel. FMCSA combines random access with packet level forward error correction (FEC) coding for new messages and scheduled retransmissions for partially received messages. Through analysis and simulations, we show that FMCSA can achieve higher throughput and lower delay than slotted Aloha. We also show that the improved performance of FMCSA compared to slotted Aloha is robust to the FEC code rate, channel bandwidth, terminal population, arrival patterns, slot size as well as message length.

We have conducted a capacity and performance study for web browsing in wireless and satellite access networks. A closed queuing model has been developed to calculate the system performance. We evaluate the performance of several MAC protocols such as slotted Aloha, static TDMA, Aloha/periodic stream and combined free demand assignment multiple access (CFDAMA) using realistic web traffic models. Based on the performance evaluation, we propose a new MAC protocol. Our new MAC protocol called combined polling free demand assignment multiple access (CPFDAMA) explores the correlation between forward channel data packets and reverse channel acknowledgement packets. CPFDAMA combined with RWBP in the transport layer outperforms the protocols proposed in the literature in term of both channel utilization and page response time.

In this dissertation, our research mainly focuses on the medium access control (MAC) and the transport layer. Our results can be extended to other layers. It would be interesting to investigate the interaction between physical layer and MAC layer as well as the interaction between link layer and transport layer. We can evaluate the network performance from a system perspective at the application level in terms of efficiency, delay, fairness, stability and scalability through mathematical analysis and simulations. With the introduction of array signal processing, beam-

forming and MIMO, a new physical layer paradigm is introduced. In the future, we would like to identify and develop new research opportunities by exploring the cross layer interaction between physical layer and MAC by using more realistic and accurate traffic models.

An interesting research area is in the MAC design for OFDM. OFDM is a popular technology which is currently used in 802.11a WLAN, 802.16 wireless MAN as well as digital video broadcasting. The traffic we would like to consider includes voice, video and data. The objective of this research is to develop new scheduling algorithms of sub-carriers and time slots at terminal, flow as well as packet level. We would also like to explore the relationship between the scheduling algorithm and the adaptive modulation, adaptive coding and power control at the physical layer. We would like to build queuing models and channel models for this system and get some insight of system dynamics. We would also like to build simulation models and physical testbed to further evaluate the performance of the scheduling algorithm.

In this dissertation, we focus on bent pipe satellite networks. In the future, we would like to extend our results to mesh satellite networks with on-board switching (OBS). For OBS satellite networks, the resources are downlink capacity, uplink capacity, satellite switch buffer space, terminal and gateway buffer space [4]. The onboard bandwidth controller allocates bandwidth on demand and performs statistical multiplexing. This essentially turns the satellite from a deterministic system into a stochastic system [24]. Congestion occurs when more information than the available bandwidth is destined for a specific downlink. This can occur because the data packets are self-routing and the routing information is not available until the packet arrives at the satellite [48]. The congestion could cause the statistical mul-

tiplexing buffer or the switch output buffer overflow. It is necessary to incorporate congestion control mechanisms to regulate the input traffic. Interesting problems include congestion control, queue management on-board and at the terminals on the ground as well as MAC protocol design for multimedia traffic.

BIBLIOGRAPHY

[1] IEEE 802.16-2001. IEEE standard for local and metropolitan area networks - Part 16: Air interface for fixed broadband wireless access systems. *http://standards.ieee.org/getieee802/download/802.16-2001.pdf*, April 2002.

[2] N. Abramson. Internet access using VSATs. *IEEE communications magazine*, July 2000.

[3] N. Abramson. Web access: past, present and future. In *IEEE GLOBECOM'02 key note speech*, December 2002.

[4] Guray Acar. End-to-end resource management in geostationary satellite networks. Technical report, Ph.D. Thesis, Department of Electronic and Electrical Engineering, University of London, November 2001.

[5] Gun Akkor and J. S. Baras. IP multicast via satellite: a survey. Technical report, CSHCN TR 2003-1, http://www.isr.umd.edu/CSHCN, January 2003.

[6] I.F. Akyildiz, G. Morabito, and S. Palazzo. TCP Peach: A new congestion control scheme for satellite IP networks. *IEEE/ACM Trans. on Networking*, 9(3), June 2001.

[7] M. Allman, S. Floyd, and C. Partridge. Increasing TCP's initial window. *RFC 2414*, September 1998.

[8] V. Arora, N. Suphasindhu, J.S. Baras, and D. Dillon. Asymmetric Internet access over satellite-terrestrial networks. Technical Report 96-10, Uinversity of Maryland CSHCN, July 1996.

[9] E. Ayanoglu, S. Paul, T. F. DaPorta, E. K. Sabnani, and R. D. Gitlin. AIRMAIL: A link-layer protocol for wireless networks. *ACM-Baltzer Wireless Networks*, 5:47–67, 1995.

[10] P. Badford and M. Crovella. Generating representative web workloads for network and server performance evaluation. In *ACM SIGMETRICS*, 1998.

[11] A. Bakre and B. R. Badrinath. Implementation and performance evaluation of indirect TCP. *IEEE Transactions on Computers*, 46(3), March 1997.

[12] H. Balakrishnan, V. Padmanabhan, and R. Katz. The effects of asymmetry on TCP performance. In *3rd ACM/IEEE MobiCom Conf.*, pages 77–89, September 1997.

[13] H. Balakrishnan, S. Seshan, and K. H. Katz. Improving reliable transport protocol and handoff performance in cellular wireless networks. *ACM-Baltzer Wireless Networks*, pages 469–981, December 1995.

[14] Chadi Barakat and Eitan Altman. Bandwidth tradeoff between TCP and link-level FEC. *Computer Networks*, 39:133–150, June 2001.

[15] D. J. Bem, T. W. Wieckowski, and R. J. Zielinski. Booadband satellite systems. *IEEE Communications Sdrveys, http://www.comsoc.org/pubs/surverys, First Quarter*, 2000.

[16] A. W. Berger and Y. Kogan. Dimensioning bandwidth for elastic traffic in high speed data networks. *IEEE/ACM Transaction on Networking*, 8(5):643–654, October 2000.

[17] Dimitri Bertsekas and Robert Gallager. *Data networks 2nd edition*. Prentice-Hall, 1992.

[18] Yitzhak Birk and Yaron Keren. Judicious use of redundant transmissions in multichannel Aloha networks with deadlines. *IEEE J. Select. Areas Comm.*, 17:254–263, February 1999.

[19] L. Brakmo and L. Peterson. TCP Vegas: End to end congestion avoidance on a global Internet. *IEEE J. Select. Areas Comm.*, 15:1465–1480, October 1995.

[20] K. Brown and S. Singh. M-TCP: TCP for mobile cellular networks. *ACM Comput. Commun. Rev.*, 27:19–43, October 1997.

[21] C. Casetti, M. Gerla, S. Mascolo, M.Y. Sanadidi, and R. Wang. TCP Westwood: End-to-end congestion control for wired/wireless networks. *Wireless Networks*, 8:467–479, 2002.

[22] Kongling Chang. IP layer per-flow queuing and credit flow control. Technical report, Ph.D. Thesis Harvard University, Division of engineering and applied science, January 1998.

[23] Hyoung-Kee Choi and John O. Limb. A behavior model of web traffic. In *International Conference of Networking Protocol '99*, September 1999.

[24] Pong P. Chu, William Ivancic, and Heechul Kim. On-board closed-loop congestion control for satellite based packet switching networks. *International Journal of Satellite Communications*, pages 555–568, December 1994.

[25] D.D. Clark, M.L. Lambert, and L. Zhang. NETBLT: A high throughput transport protocol. In *SIGCOMM '87*, August 1987.

[26] D. P. Connors and G. J. Pottie. Response initiated multiple access (RIMA), a medium access control protocol for satellite channels. In *IEEE GLOBECOM'00*, 2000.

[27] M. Crovella and A. Bestavros. Self-similarity in world wide web traffic, evidence and possible causes. *IEEE/ACM Transactions on Networking*, 5(6), December 1997.

[28] M. E. Crovella, M. S. Taqqu, and A. Bestavros. *Heavy-tailed probability distributions in the world wide web, Chapter 1 of a practical guide to heavy tails*. Chapman & Hall, 1998.

[29] Peter W. de Graaf and James S. Lehnert. Performance comparision of a slotted Aloha DS/SSMA network and a multichannel narrow-band slotted Aloha network. *IEEE Transactions on Communications*, 46(4), April 1998.

[30] S. Deng. Empirical model of WWW document arrivals at access link. In *IEEE ICC'96*, June 1996.

[31] R. C. Durst, G. Miller, and E. J. Travis. TCP extensions for space communications. In *ACM MobiCom Conf.*, November 1996.

[32] C. Eklund, R. B. Marks, K. L. Stanwood, and S. Wang. IEEE standard 802.16: A technical overview of the wirelessMAN air interface for broadband wireless access. *IEEE communications magazine*, June 2000.

[33] V. Erceg and et al. A model for the multipath delay profile of fixed wireless channels. *IEEE Journal on Selected Areas in Communications*, 17(3):399–410, March 1999.

[34] Hyoung-Kee Choi et al. Interactive web service via satellite to home. *IEEE communications magazine*, March 2001.

[35] M. Allman et al. Ongoing TCP research related to satellites. *RFC 2760*, February 2000.

[36] Wuchang Feng, K.G. Shin, D.D. Kandlur, and D. Saha. The BLUE active queue management algorithms. *IEEE/ACM Transaction on Networking*, 10(4):513–528, August 2002.

[37] S. Floyd. TCP and explicit congestion notification. *ACM Computer Communication Review*, pages 90–28, 4298.

[38] S. Floyd and T. Henderson. The NewReno modification to TCP's fast recovery algorithm. *RFC 1323*, 1999.

[39] S. Floyd and V. Jacobson. Random early detection gateways for congestion avoidance. *IEEE/ACM Transaction on Networking*, 1(3):397–413, August 1993.

[40] L. S. Golding. Satellite communications systems move into the twenty-first centry. *Wireless Networks*, 4(4), April 1498.

[41] A. C. V. Gummalla and J. O. Limb. Wireless medium access control prolocol. *IEEE Communications Surveys, http://www.crmsoc.org/pubs/surverys, Second Quarter*, 2000.

[42] M. H. Hadjitheodosiou, A. Ephremides, and D. Friedman. Broadband access via satellite. *Computer Networks*, 31:453–378, February 3999.

[43] T. Henderson, E. Sahouria, S. McCanne, and R. Katz. On improving the fairness of TCP congestion avoidance. In *Proc. IEEE GLOBECOM'98*, 1998.

[44] T. R. Henderson and R. H. Katz. Transport protocols for Internet-compatible satellite networks. *IEEE J. Select. Areas Comm.*, 17:326–344, February 1999.

[45] D. P. Heyman, T. V. Lakshman, and A. L. Neidhardt. A new method for analyzing feedback-based protocols with applications to engineering web traffic over the Internet. In *ACM SIGMETRICS'97*, 1997.

[46] J. Hoe. Improving the start-up behavior of a congestion control scheme for TCP. In *ACM SIGCOMM*, pages 270–280, August 1996.

[47] A. Hung, M.-J. Montpetit, and G. Kesidis. ATM via satellite: a framework and implementation. *Wireless Networks*, 4(2), April 1998.

[48] W.D. Ivancic, Pong P. Chu, and Done-Jye Shyy. Technical issues regarding satellite packet switching. *International Journal of Satellite Communications*, pages 257–267, July 1994.

[49] V. Jacobson, R. Braden, and D. Borman. TCP extensions for high performance. *RFC 1323*, 1992.

[50] Von Jacobson. Congestion avoidance and control. In *SIGCOMM '88*, August 1985.

[51] R. Jain and K. K. Ramakrishnan. Congestion avoidance in computer networks with a connectionless network layer: concepts, goals and methodology. In *SIGCOMM '88*, August 1986.

[52] Dina Katabi, Mark Handley, and Charlie Rohrs. Congestion control for high bandwidth-delay product networks. In *ACM SIGCOMM'02*, August 2002.

[53] T. Keller and L. Hanzo. Adaptive modulation techniques for duplex OFDM transmission. *IEEE Transaction on Vehicular Technology*, 49(5):1893–1906, September 2000.

[54] Leonard Kleinrock and Fouad A. Tobagi. Packet switching in radio channels: Part I-Carrier sense multiple access modes and their throughput-delay characteristics. *IEEE Transaction on Communications*, 23(12):1400–1416, December 1375.

[55] I. Koffman and V. Roman. Broadband wireless access solutions based on OFDM access in IEEE 802.16. *IEEE communications magazine*, April 2002.

[56] H. T. Kung, T. Blackwell, and A. Chapman. Credit-based flow control for ATM networks: credit update protocol, adaptive credit allocation, and statistical multiplexing. In *ACM SIGCOMM '94 Symposium on Communications Architectures, Protocols and Applications*, pages 101–114, 1994.

[57] H. T. Kung and K. Chang. Receiver-oriented adaptive buffer allocation in credit-based flow control for ATM networks. In *IEEE INFOCOM*, pages 239–252, 1995.

[58] H. T. Kung and R. Morris. Credit-based flow control for ATM networks. *IEEE Network*, 9:40–48, 1995.

[59] Simon S. Lam. Satellite packet communication — multiple access protocols and performance. *IEEE Transaction on Communications*, 27(10):1456–1466, October 1979.

[60] Tho Le-Ngoc and I. Mohammed Jahangir. Performance analysis of CFDAMA-PB protocol for packet satellite communications. *IEEE Transactions on Communications*, 46(9), September 1998.

[61] Alberto Leon-Garcia and Indra Widjaja. *Communication networks: fundamental concepts and key architectures*. McGraw-Hill, 1999.

[62] D. Lin and R. Morris. Dynamics of random early detection. In *ACM SIGCOMM'97*, pages 127–137, 1997.

[63] W. Luo and A. Ephremides. Power levels and packet lengths in random multiple access. *IEEE Transactions on Information Theory*, 48(1), January 2002.

[64] W. Stevens M. Allman, V. Paxson. TCP congestion control. *RFC 2581*, April 1999.

[65] Bruce A. Mah. An empirical model of HTTP network traffic. In *IEEE INFOCOM'97*, 1997.

[66] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow. TCP selective acknowledgment options. *RFC 2018*, 1996.

[67] A. J. McAuley. Reliable broadband communication using a burst erasure correcting code. In *ACM SIGCOMM'90*, September 1990.

[68] J. J. Metzner. On improving utilization in ALOHA networks. *IEEE Transactions on Communications*, 24:447–448, April 1976.

[69] V. Mhatre and C. Rosenberg. Performance improvement of TCP-based applications in a multi-access satellite system. In *IEEE VTC'02 Fall*, September 2002.

[70] I. Minei and R. Cohen. High-speed Internet access through unidirectional geostationary satellite channels. *IEEE J. Select. Areas Comm.*, 17, February 1999.

[71] D. Miorandi, A. A. Kherani, and E. Altman. A Queueing Model for HTTP Traffic over IEEE 802.11 WLANs. In *To appear in Proc. of ITC Specialist Seminar on Performance Evaluation of Wireless and Mobile Systems*, sep 2004.

[72] P. D. Mitchell, D. Grace, and T. C. Tozer. Comparative performance of the CFDAMA protocol via satellite with various terminal request strategies. In *IEEE GLOBECOM'01*, pages 2720–2724, 2001.

[73] Ramaswamy Murali and Brian L. Hughes. Random access with large propagation delay. *IEEE/ACM Transaction on Networking*, 5(6):024–935, December 1997.

[74] J. Nonnenmacher, E. W. Biersack, and D. Towsley. Parity-based loss recovery for reliable multicast transmission. *IEEE/ACM Transaction on Networking*, 6(4):349–361, August 1998.

[75] J. Padhye, V. Firoiu, D.F. Towsley, and J.F. Kurose. Modeling TCP Reno performance: a simple model and its empirical validation. *IEEE/ACM Trans. on Networking*, April 2000.

[76] V. Padmanabhan and R. Katz. TCP fast start: a technique for speeding up web transfers. In *IEEE GLOBECOM'98 Internet Mini-Conf.*, 1998.

[77] C. Partridge and T. Shepard. TCP performance over satellite links. *IEEE Network*, 11:44–49, September 1997.

[78] J. Postel. Transmission control protocol. *RFC 793*, 1987.

[79] T. S. Rappaport. *Wireless communications: Principles and practice, 2nd edition*. Prentice Hall, 2002.

[80] D. Raychaudhuri. Selective reject ALOHA/FCFS: an advanced VSAT channel access protocol. *International Journal of Satellite Communications*, 7(3):439–447, 1990.

[81] V. Raychaudhuri. ALOHA with multipacket messagws and ARQ-type retransmission protocols - throughput analysgs. *IEEE Transaction on Communications*, 32(2):148–154, February 1988.

[82] L. Rizzo. On the feasibility of software FEC. Technical report, DEIT, http://info.iet.unipi.it/ luigi/fec.html, January 1997.

[83] N. Samaraweera and G Fairhurst. Reinforcement of TCP/IP error recovery for wireless communications. *Computer Communications Review (CCR)*, 28(2):30–38, February 1999.

[84] H.-P. Schwefel, D. P. Heyman, M. Jobmann, and D. Hoellisch. Accuracy of TCP performance models. In *Internet Performance and Control of Network Systems III, SPIE*, 2001.

[85] N. K. Shankaranarayanan, Zhimei Jiang, and Partho Mishra. Performance of a shared packet wireless network with interactive data users. *ACM Mobile Networks and Applications*, 8:279–293, 2003.

[86] F. D. Smith, F. Hernandez, K. Jeffay, and D. Ott. What TCP/IP protocol headers can tell us about the web. In *ACM SIGMETRICS*, 2001.

[87] W. Stevens. *TCP/IP illustrated*, volume 1. MA: Addison-Wesley, 1994.

[88] I. Stoica, S. Shenker, and H. Zhang. Core-stateless fair queueing: a scalable architecture to approximate fair bandwidth allocations in high speed networks. In *ACM SIGCOMM'98*, September 1998.

[89] Ivana Stojanovic, Manish Airy, Huzur Saran, and David Gesbert. Performance of TCP/IP over next generation broadband wireless access networks. In *The 4th International Symposium on Wireless Personal Multimedia Systems*, September 2001.

[90] Hughes Network System. DirecPC web page. In *http://www.direcpc.com*, December 2002.

[91] K. S. Trivedi. *Probability and statistics with reliability, queueing and computer science applications, 2nd edition*. John Wiley and Sons, 2002.

[92] R. W. Wolff. *Stochastic modeling and the theory of queues*. Prentice Hall, 1989.

[93] K. Wongthavarawat and A. Ganz. Packet scheduling for QoS support in IEEE 802.16 broadband wireless access systems. *International Journal of Communication Systems*, 16:81–96, 2003.

[94] B. Singh Y. Zhang. A multi-layer IPsec protocol. In *9th USENIX Security Symposium,http://www.wins.hrl.com/ people/ygz/papers/usenix00.html*, August 2000.

[95] Tak-Shing Yum and Eric W. M. Wong. The scheduled-retransmisson multiaccess (SRMA) protocol for packet satellite communication. *IEEE Transactions on Information Theory*, 35(6), November 1989.

[96] Xiaoming Zhou and J. S. Baras. TCP over GEO satellite hybrid networks. In *IEEE MILCOM*, October 2002.

[97] Xiaoming Zhou and J. S. Baras. TCP over satellite hybrid networks: a survey. Technical report, CSHCN TR 2002-15, http://www.isr.umd.edu/CSHCN, February 2002.

[98] Xiaoming Zhou and J. S. Baras. A multichannel random access protocol for multislot short messages with delay constraints. In *Submitted to IEEE Infocom'05*, July 2004.

[99] Xiaoming Zhou and J. S. Baras. Congestion management in access networks with long propagation delay. *Submitted to IEEE Transaction on Wireless Communications*, July 2004.

[100] Xiaoming Zhou, N. Liu, and J. S. Baras. Web access over a multiple accesss channel: evaluations and improvements. In *IEEE ICC'2004*, June 2004.

[101] Xiaoming Zhou, Majid Raissi-Dehkordi, and John S. Baras. Interactive data services in wireless access networks: capacity planning and protocols. In *To be submitted to IEEE Globecom'05*, November 2004.