INVESTIGATION OF REACTIVE TCP AND LINK CHARACTERISTICS ESTIMATION FOR WIRELESS LINKS

WU XIUCHAO

NATIONAL UNIVERSITY OF SINGAPORE

2004

INVESTIGATION OF REACTIVE TCP AND LINK CHARACTERISTICS ESTIMATION FOR WIRELESS LINKS

BY

WU XIUCHAO

(*B.E., USTC, PRC*)

A THESIS SUBMITTED
FOR THE DEGREE OF MASTER OF SCIENCE
DEPARTMENT OF COMPUTER SCIENCE
SCHOOL OF COMPUTING
NATIONAL UNIVERSITY OF SINGAPORE

2004

To my parents

Acknowledgements

During my work on Reactive TCP and Wireless Link Characteristics Estimation, many people have contributed valuable help and advices. Without their kind assistance, I could not finish this project smoothly.

I would first thank my supervisor, Associate Professor A.L. Ananda, for his clear guidance and faith in letting me pursue these new directions. From him I learned how to do research and how to think when encountering problems. I can not thank him enough for all what he has done for me.

I must express my sincere gratitude to Dr. Chan Mun Choon, Dr. Samarjit Chakraborty, and Dr. Rajeev Shorey. They gave me invaluable advices during my research and/or thesis examination. I must also thank Venkatesh S. Obanaik, Indradeep Biswas, Shao Tao, and Wang Yu. As collaborators, they gave me a lot of help in work.

Last but not the least, I should thank Fu Qian, Dou Qinfeng, Aurbind Sharma, and other friends, who have aided me in one way or another.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

ABW	Available Bandwidth
ACK	Acknowledgement
AIMD	Additive Increase Multiply Decrease
AP	Access Point
ARQ	Automatic Repeat reQuest
BDP	Bandwidth Delay Product
BER	Bit Error Rate
CA	Congestion Avoidance
CW	Contention Window
CWND	Congestion Window
CTS	Clear To Send
DCF	Distributed Coordination Function
DIFS	Distributed Inter Frame Spacing

ECC	Error Correction Code
FEC	Forward Error Correction
FER	Frame Error Rate
FTP	File Transfer Protocol
GEO	Geo-stationary Earth Orbit
GPRS	General Packet Radio Service
IP	Internet Protocol
IR	Infrared
LEO	Low Earth Orbit
LFN	Long Fat Network
LTN	Long Thin Network
MAC	Medium Access Control
MSS	Maximum Segment Size
NS	Network Simulator
PCF	Point Coordination Function
PDA	Personal Digital Assistant
PLCP	Physical Layer Convergence Procedure
PLR	Packet Loss Rate

PMD	Physical Medium Dependent
RTS	Request To Send
RTT	Round Trip Time
RTTM	Round Trip Time Measurement
SACK	Selective Acknowledgement
SIFS	Short Inter Frame Spacing
SNR	Signal to Noise Ratio
SS	Slow Start
SSTHRESH	Slow Start Threshold
TCP	Transmission Control Protocol
WLAN	Wireless Local Area Network
WMAN	Wireless Metropolitan Area Network
WWW	World Wide Web

Summary

TCP, perhaps the most widely used transport protocol, was designed for highly reliable links and stationery hosts. The characteristics of wireless links, lossy and mobility, undermine the assumptions of TCP protocol. Since more and more people use wireless links to access Internet and Intranet, it is worthwhile to improve TCP performance over wireless links.

Many mechanisms of TCP have been proposed in order to solve problems brought by wireless links. But the dynamics of wireless links and potential vertical handoff among multiple interfaces installed on a mobile node give different network path characteristics to a TCP connection at different time. The changing link characteristics pose different problems to TCP, thus different mechanisms are necessary to handle them at different time. It is impossible to use a fixed set of TCP mechanisms to achieve optimal performance over wireless links. Reactive TCP, which adopts different mechanisms according to different network path characteristics, should be a useful method to improve TCP performance over wireless links.

Network path characteristics, which enable Reactive TCP to function, should be estimated accurately and timely in order to assure the success of Reactive TCP. Due to fading, mobil-

ity, and possible contention among mobile nodes, the characteristics of a wireless link may change frequently and abruptly. Commonly used probing-packets methods are not appropriate for a network path with a wireless link because they could not estimate network path characteristics accurately and timely with small cost.

Based on the fact that wireless link is commonly the last link, the bottleneck, and the most dynamic link, it often determines the characteristics of a network path. So it is still valuable for Reactive TCP to estimate the characteristics of wireless links. In addition, Access Point or Base Station can know all communications over a wireless network. From these knowledge, AP can deduce contention status of the wireless network. With contention status from AP and the quality of its wireless link, a mobile node can estimate the wireless link characteristics experienced by itself.

In this thesis, we first design an architecture for Reactive TCP, analyze the functions of TCP protocol, discuss how to react to network path characteristics, and propose a protocol framework to support Reactive TCP with multiple interfaces. We then propose a new non-intrusive mechanism to estimate link characteristics of IEEE 802.11 DCF based WLAN, one of the most popular wireless access networks. Through simulation experiments, we find that it is possible to estimate wireless link characteristics accurately and timely with small cost. These works pave the way for the future work in this environment.

Chapter 1

Introduction

1.1 TCP Protocol

In recent years, Internet and Intranet (the internet within an organization), have achieved huge success. More and more tasks of daily life and business are being carried out over the Internet and Intranet. TCP/IP is the cornerstone of Internet and Intranet. IP (Internet Protocol) [39] is the glue which holds heterogeneous networks together and provides necessary functions to transfer packets over these networks. TCP (Transmission Control Protocol) [40] provides a connection-oriented end-to-end service and ensures the reliable and in-order transfers of data. Over the years, TCP has facilitated the development of various applications (FTP, TELNET, WWW, etc.) which are responsible for the success of Internet and Intranet.

TCP is an end-to-end transport protocol. There are exactly two endpoints on a TCP connection. They use sliding window mechanism to transmit data. After a TCP connection is established by three-way handshake, the sender begins to send data in segments whose maximum size is negotiated during handshake, and each byte sent by the sender has a sequence number. The sender continues to send all segments permitted by its sending window. When the receiver gets new segments, it sends back an ACK packet, which contains sequence number of the next expected byte, to open window for the sender. When the sender gets a new ACK packet, it slides its sending window, discards acknowledged data, and begins to transmit new data which is permitted to be sent after sliding.

The sending window is determined by congestion control of the sender and flow control of the receiver. Congestion Window (CWND) is a parameter maintained by TCP sender for congestion control. In congestion control, the sender probes for a data rate as high as possible by increasing CWND continuously and recovers from congestion by decreasing CWND when congestion is detected. The sender regards the loss of a segment as a signal of congestion and recovers from the loss with go-back-N retransmission mechanism. In flow control, the receiver gives TCP sender an advertisement window (WND) in ACK packet according to its buffer. The sending window is the smaller one of CWND and WND.

In 1980s, TCP was designed for highly reliable links and stationery hosts. It faces many problems when communication links with different characteristics are used. For example, TCP can not fully utilize bandwidth provided by a Long Fat Network (LFN) [41] if its Bandwidth-Delay Product (BDP) exceeds the range of advertisement window (16-bits)of TCP header. Especially, TCP faces many serious problems when it is used over wireless links. In the next section, we first introduce the characteristics of wireless links.

1.2 Wireless Links

Normally, the network path used by a TCP connection can be divided into two parts — Core Network and Access Network. Core Network is comprised by high speed routers and optical fiber links. Access Network connects users to Core Network. Different communication links can be used as Access Network. Normally, the path characteristics are dominated by the access link.

With the development of wireless communication, it is reasonable to use wireless links to access Internet&Intranet because these links enable the mobile and cordless Internet&Intranet access. In order to utilize existing applications, it is a straightforward approach to use TCP over wireless links.

A lot of different wireless links, such as GSM-CSD [51], GSM-HSCSD, WaveLAN, GPRS [3], WCDMA [1], IEEE 802.11 [5, 6, 7], Bluetooth [10], and Satellite links, have been used to access Internet and Intranet. For example, GPRS may be used to access email by mobile users and IEEE 802.11 may be used as Ethernet in an corporation to access Intranet.

These wireless links own very different characteristics that pose different problems to TCP. For example, TCP suffers frequent segment loss due to transmission error over wireless links without FEC and ARQ, such as WaveLAN. These lossy wireless links undermine the assumption of TCP that the loss of segment is caused by congestion. TCP sender will reduce its sending rate unnecessarily and result in poor throughput. As for wireless links with

Table 1.1: Characteristics of Wireless Links

Category	Network	Bandwidth	Link Layer	Delay	Jitter
WLAN	WaveLAN	2Mbps		Short	Small
	IEEE802.11	2Mbps	ARQ(MAC)	Short	Large
	IEEE802.11b	11Mbps	ARQ(MAC)	Short	Large
	IEEE802.11a/g	54Mbps	ARQ(MAC)	Short	Large
	Bluetooth	1Mbps	FEC	Short	Small
WMAN	GSM-HSCSD	64kbps	ARQ (RLP, FEC)	Medium	Large
	GPRS	172Kbps	ARQ(RLP)	Medium	Large
	W-CDMA	2Mbps	ARQ(RLC)	Medium	Large
Satellite	GlobalStar(LEO)	9.6Kbps	FEC	Long	Small
	ARCS(GEO)	2Mbps(U) 45Mbps(D)	FEC	Very Long	Small

ARQ, such as GPRS, local retransmission will result in large delay variation to TCP. Large delay variation may cause spurious timeout [35] in TCP and results in reduced throughput.

According to their range, wireless links can be classified into Wireless LAN (WLAN), Wireless MAN (WMAN), and Satellite. Table 1.1 summaries the characteristics of several widely used wireless links whose bandwidth vary from several Kbps to tens Mbps.

1.3 Thesis Motivation

TCP performance enhancement is an active research area. Several TCP implementations (Tahoe, Reno, New Reno) have been approved. New Reno [33] is the latest approved implementation. Many other TCP implementations have also been proposed, such as Vegas [27, 28], Westwood [29], etc. In addition to these TCP implementations, many mechanisms have been proposed to solve problems posed by different link characteristics. Dawkins [31, 32] summarizes TCP performance issues over slow links and lossy links. Balakrishnan [20] summarizes TCP performance issues over asymmetric links. Allman [14] investigates into enhancing TCP performance over satellite links. The 2.5G and 3G wireless links are investigated in [36].

But all these works have not considered the changing network path characteristics that a TCP connection may experience, especially when wireless link(s) is(are) used. In the next subsection, we describe the motivations for Reactive TCP over wireless links and wireless link characteristics estimation.

1.3.1 Motivation for Reactive TCP

Different wireless links provide services of different bandwidth, coverage, price, etc. No single wireless communication technology can simultaneously provide a low-latency, high-bandwidth, wide-area data service to a large number of mobile users [47]. Mobile devices with multiple interfaces can utilize the most appropriate wireless link and provide the best service to the user. For example, considering a personal data assistant (PDA) installed with IEEE 802.11 and GPRS interfaces, it can get high bandwidth in office through IEEE 802.11 interface. When the user moves out of the range of IEEE 802.11 based WLAN, it can still maintain the access to Internet through GPRS interface.

Currently, more and more mobile devices have been installed with multiple wireless interfaces. Intel plans to support Wi-Fi, Wi-MAX, and WCDMA in its next generation Centrino CPU. With support from mobile IP, a TCP connection may survive through multiple interfaces. That means a TCP connection will experience different characteristics of different interfaces. A TCP connection must handle all problems posed by different interfaces.

Even though only one wireless interface is used, TCP still suffers different characteristics of the wireless link at different time. Compared with wired links, the most outstanding characteristic of wireless communication is the high Bit Error Rate (BER) of a wireless channel. BER of a wireless link is determined by its link quality which may vary frequently and abruptly due to fading, handoff, multi-path, etc. BER determines Packet Loss Rate (PLR) and affects Available Bandwidth (ABW). If ARQ is used in link layer, delay experienced by TCP is also affected by BER. In addition, in WLAN, available bandwidth will also be affected by contention among nodes. Thus, a wireless link gives TCP different link characteristics at different time.

In a nutshell, a TCP connection need to handle different problems posed by wireless link(s) at different time. It is impossible to use a fixed set of algorithms, which are proposed for specific problems posed by specific link characteristics (particularly wired links), to achieve optimal performance in the wireless domain. Reactive TCP, which adjusts its algorithms according to current network path characteristics, is a feasible solution and worthwhile to be investigated.

1.3.2 Motivation for Wireless Link Characteristics Estimation

Network path characteristics form the input to Reactive TCP. Reactive TCP adjusts its behaviors according to current network path characteristics. Algorithms, which can estimate network path characteristics accurately and timely, are necessary for the success of Reactive TCP.

Not only Reactive TCP, other adaptation protocols can also benefit from the knowledge of network path characteristics. For example, rate-based streaming applications [30] can adjust coding scheme based on available bandwidth to achieve optimal stream quality. They can also set buffer size according to delay variation and stream data rate in order to handle stream jitter and avoid wasting memory. It is really very valuable to estimate network path characteristics accurately and timely.

Network path characteristics can be estimated according to the status of internal routers or estimated at end points. Due to the unwieldy complexity of maintaining status per connection at internal routers, network path characteristics are normally estimated at end points. Many algorithms, such as Delphi [55], pathload [44], pathchar [43], and pathChirp [56], have been proposed to estimate network path characteristics — especially ABW. They send probing-packets and estimate network path characteristics by analyzing delay experienced by these probing-packets. They are intrusive estimation algorithms because probing-packets consume bandwidth of the network path.

Intrusive estimation algorithms are not appropriate when a wireless link is used in a network path. Because of the precious bandwidth of wireless link, high dynamic wireless link quality due to fading and mobility, and possible contention among nodes of wireless network, intrusive algorithms could not estimate network path characteristics accurately and timely with small overhead of probing-packets. But without probing-packets, the end points can not get the status of internal routers. That means it is very hard to estimate network path without support from internal routers and probing-packets.

Currently, a wireless link is normally used as the access link (the last link of a network path), and it normally dominates the characteristics of a network path. Firstly, compared with other wired links of core network, the bandwidth of a wireless link is much lower and more dynamic due to mobility, fading and contention. Wireless link is normally the bottleneck link and determines available bandwidth of a network path. Nextly, a wireless link has much higher BER than wired links and determines packet loss rate (PLR) of a network path. Lastly, because of its high BER and local retransmission commonly used over wireless link, delay variation of a wireless link could dominate delay variation of a network path.

Based on above facts, non-intrusive algorithms to estimate wireless link characteristics at the end point (mobile node) are valuable.

1.4 Thesis Contributions

In this thesis, we investigate into Reactive TCP. We first propose an architecture for Reactive TCP. We then analyze TCP protocol, especially its congestion control mechanism. We also summarize different problems posed by different link characteristics and corresponding TCP enhancements. This survey can guide Reactive TCP about how to react to link characteristics. After that, we propose a protocol framework to support Reactive TCP with multiple interfaces.

Algorithms, which can estimate wireless link characteristics accurately and timely without high cost, are necessary for the success of Reactive TCP over wireless networks. The change of link characteristics due to vertical handoff [47] can be coarsely estimated by TCP according to current interface used by a node. It is more difficult to estimate link characteristics of the same link, which changes frequently and abruptly due to mobility, fading and potential contention. In this thesis, we propose a new non-intrusive link characteristics estimation mechanism for IEEE 802.11 DCF based WLAN, one of the most popular wireless access networks. Instead of sending probing-packets, a mobile node estimates its link characteristics based on wireless link quality and contention status of the whole WLAN.

In NS2, we implement this WLAN link characteristics estimation mechanism and the protocol framework proposed to support Reactive TCP over multiple interfaces. We also test the accuracy of our WLAN link characteristics estimation mechanism through simulation experiments.

1.5 Thesis Walkthrough

This Thesis is organized as follows.

Chapter 2 investigates into Reactive TCP. We propose an architecture of Reactive TCP, analyze TCP protocol, and summarize problems posed by different link characteristics and their corresponding solutions. We also proposed a framework to support Reactive TCP with multiple interfaces.

Chapter 3 presents a new non-intrusive link characteristics estimation mechanism for IEEE 802.11 DCF based WLAN, one of the most popular wireless access networks.

Chapter 4 describes how to simulate our WLAN link characteristics estimation mechanism in NS2. We first present how to simulate a WLAN channel in office environment. We then describe how to implement our mechanism in NS2.

Chapter 5 presents several experiments designed to test the accuracy of our link characteristics estimation mechanism for IEEE 802.11 DCF based WLAN. We also analyze and discuss their results.

Chapter 6 summarizes the work that has been done in this thesis project, and finally draws our conclusion with some future works.

Chapter 2

Reactive TCP

Since many links with different characteristics, especially dynamic wireless links, are used in Internet and Intranet, a TCP connection may suffer different problems, caused by different network path characteristics, at different time.

In addition, there are many applications based on TCP. These applications have different expectations from TCP. For example, Telnet expects short response time, but FTP expects high throughput. Thus, Nagle algorithm [52] which avoids to send short packets should be enabled for FTP and disabled for Telnet.

Moreover, the TCP endpoints may be used in different environments, which have different constrains. For example, when a notebook works outside of office, TCP should try to avoid unnecessary retransmission to save the limited power of the battery. When it is used in office with power supply, TCP need not unduly worry about this.

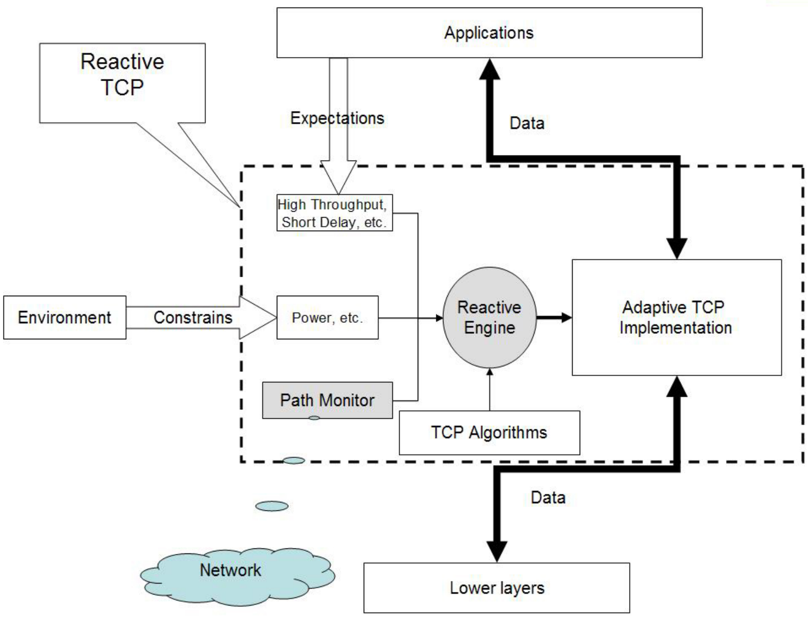

Figure 2.1: Reactive TCP Architecture

All these factors make Reactive TCP, which adjusts its behaviors according to current network path characteristics, application expectations, and environment constrains, a prospective solution. Figure 2.1 depicts our architecture proposed for Reactive TCP.

TCP algorithms are those algorithms which have been proposed for different links. Reactive TCP does not propose new algorithms for any link. It just utilizes the most appropriate existing algorithms to enhance TCP performance. Path monitor estimates network path characteristics and reports current path characteristics to Reactive Engine. Reactive Engine accepts input (application expectations, environment constrains, and current network path characteristics) and selects a proper set of algorithms for TCP functions. According to the output of Reactive Engine, Adaptive TCP Implementation can change algorithms used by a TCP connection at any time. Thus a TCP connection can use proper algorithms, which are selected by Reactive Engine according to current network path characteristics, application expectations, and environment constrains, to improve TCP performance.

Reactive Engine is the core of Reactive TCP. It is responsible to select algorithms according to current network path characteristics, application expectations, and environment constrains. During the life of a TCP connection, application expectations and environment constrains will not change frequently. And their effects are easy to understand. Network path characteristics have many metrics. They may change frequently and pose different challenges to TCP. The analysis of network path characteristics and their effects to TCP is necessary to implement Reactive Engine. A number of papers have discussed TCP mechanisms for different links, such as asymmetric links [20], lossy links [32], slow links [31], 2.5G-3G wireless links [36], satellite links [14], LTN [50], LFN[41], etc. Since the characteristics of access link normally determines network path characteristics, these work provide a solid base to Reactive Engine.

In this chapter, we investigate how Reactive TCP should react to network path characteristics. Firstly, we analyze TCP protocol, especially its congestion control mechanism which affects TCP performance very much, in order to understand functions of TCP protocol. This work is helpful to design an Adaptive TCP Implementation. Secondly, we summarize network path characteristics, their effects on TCP, and TCP algorithms proposed for different link characteristics. This work is useful to design rules used by Reactive Engine. Thirdly, we propose a framework to support Reactive TCP with multiple interfaces.

2.1 TCP Analysis

According to TCP protocol, the sender is responsible to send data as fast as possible and avoid congestion collapse. The receiver is responsible to acknowledge data received by it and carry out flow control.

There are many different implementations of TCP sender and receiver, such as Tahoe, Reno, New Reno, etc. They may use different algorithms for an identical function. In this section, we will analyze the functions which should be implemented by TCP sender and receiver.

TCP receiver is quite simple. It just needs to decide what to be send in ACK packet and when to send ACK packet. In standard implementation, except WND used for flow control, ACK only includes the sequence number of the next expected byte. And the receiver sends back an ACK after two segments have been received.

TCP sender is much more complex than TCP receiver. It need to probe available bandwidth, detect or avoid congestion, retransmit lost segments, and recover from congestion. The following subsections analyze approved TCP sender implementations and TCP Vegas, a new design of TCP.

2.1.1 Approved TCP Implementations—Tahoe, Reno, and New Reno

These implementations increase CWND to probe available bandwidth. Congestion may occur sometimes if a connection lives long enough and has enough data to be sent. TCP regards the loss of segment as the signal of congestion. When segment loss is detected, these implementations retransmit lost segment and reduce CWND to recover from congestion.

In order to probe available bandwidth as soon as possible and avoid frequent congestion, TCP sender has two states, Slow Start (SS) and Congestion Avoidance (CA) [15]. A variable, Slow Start Threshold (SSTHRESH), is maintained to determine the state of a TCP sender. When CWND is less than SSTHRESH, the sender is in SS state. Otherwise, it is in CA state. SSTHRESH is set to 65535 initially.

1. Slow Start (SS): In SS state, CWND is increased by one when one ACK is received. Thus, CWND is increased exponentially. This will help the sender arrive high sending rate soon so that the sender can probe network available bandwidth quickly. The initial value of CWND is set to one.

2. Congestion Avoidance (CA): When CWND is larger than SSTHRESH, the sender enters into CA state. CWND is increased by one segment per RTT so that the sender can still probe network resource but will not cause congestion too frequently.

Since TCP sender needs to create congestion in order to probe available bandwidth, timely and accurate detection of congestion is very important to TCP sender. The following mechanisms have been proposed to detect the loss of a segment, the signal of congestion.

1. Timeout: There is a Retransmission Timer (RTO) in the sender. If RTO has expired since a segment was sent and its ACK has not received yet, the sender assumes that the segment has been lost. RTO is calculated from the mean and variance of Round Trip Time (RTT). RTT is monitored by the sender through measuring the time from

sending segment to receiving corresponding ACK.

2. Fast Retransmission [15]: With Timeout, at least a RTO is needed to detect a segment loss. RTO is relatively too long for the sender to recover from the loss quickly. The problem is even worse over links with long delay. Fast Retransmission is proposed for this problem. When a new out-of-order segment is received, TCP receiver sends back a duplicate ACK, whose expected sequence number is identical to that of previous ACKs. Fast Retransmission assumes that the offset of out-of-order segments is normally less than three. So, it regards three duplicate ACKs as the signal that a segment has been lost.

When a segment is lost and the loss is detected, the sender will retransmit the lost segments. In standard, the sender uses go-back-N retransmission mechanism. That means the lost segment and its following segments are all retransmitted.

Not only TCP sender must do retransmission, but also it must reduce sending rate in order to recover from congestion. SSTHRESH is always set to half of current CWND. But different algorithms have been proposed to do congestion recovery. These algorithms differ in Tahoe, Reno, and New Reno implementations.

1. Tahoe: When segment loss is detected through Timeout or Fast Retransmission, CWND is always set to one and TCP sender enters into SS state.

2. Reno: If Timeout occurs, CWND is set to one and TCP sender enters into SS state. If the loss is detected by Fast Retransmission (three duplicate ACKs), Reno sender retransmits the lost segment and enters into Fast Recovery [15]. Duplicate ACKs in-

dicate not only that a segment is lost but also that there is still data flowing between the two ends. To avoid an abrupt reduction of sending rate, SSTHRESH is set to half of current CWND and CWND is set to SSTHRESH+3. After that, CWND is increased by one segment for each duplicate ACK. When new ACK, which acknowledges all data sent before retransmission, is received, CWND is set to SSTHRESH and the sender returns back to CA state.

3. New Reno: New Reno is very similar to Reno. The difference is in Fast Recovery state. In Reno, if multiple segments are lost, the sender can not transfer from Fast Recovery to CA state. It will wait for one Timeout and enter into SS state. This will hurt TCP performance. In New Reno, the sender will retransmit other lost segments which are detected by three duplicate partial ACKs (these ACK packets acknowledge partial data which had been sent before the first loss was detected), and CWND will not be reduced again. This algorithm can improve TCP performance when multiple segment loss occurs frequently in one sending window. Figure 2.2 (next page) shows the congestion control of New Reno.

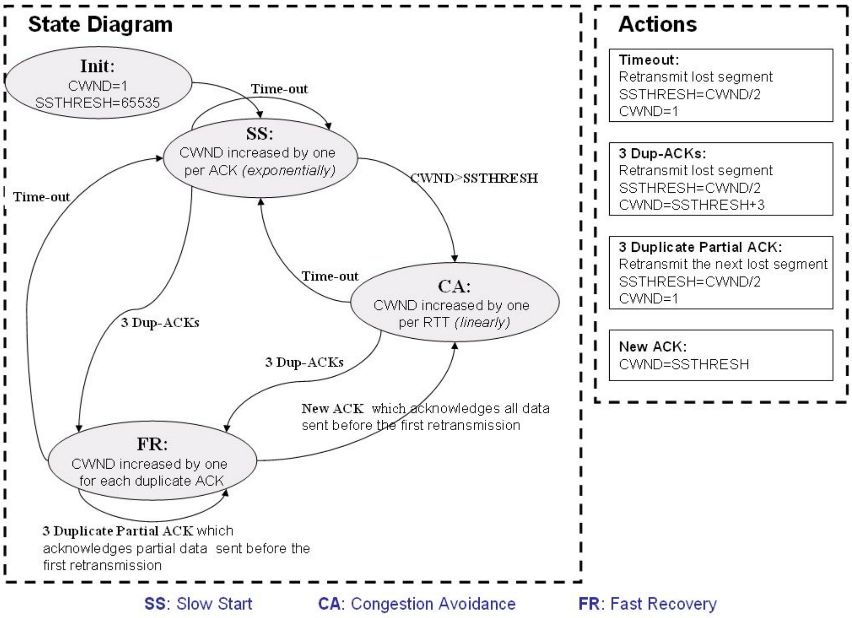

Figure 2.2: Congestion Control of TCP New Reno

2.1.2 TCP Vegas

TCP Vegas [27, 28] is a new design of TCP. Instead of reactive to congestion as TCP Reno, TCP Vegas is proactive. TCP Reno need create segment loss to detect congestion and recover from congestion. It has no mechanism to detect the forthcoming congestion before a segment is lost and hence can not prevent such loss. As for TCP Vegas, it tries to sense the forthcoming congestion by observing changes of the throughput rate. TCP Vegas adjusts CWND based on this measurement so that it can reduce the sending rate before the connection experiences segment loss.

TCP Vegas uses an aggressive retransmission mechanism, an innovative congestion avoidance mechanism, and a modified slow start mechanism.

1. Aggressive Retransmission : TCP Vegas measures RTT per segment through Time Stamp Option and calculates RTO for each segment. When it receives a duplicate ACK, it checks whether the segment's retransmission timer has expired. If so, the segment is retransmitted and congestion window is decreased. After that, when the first or second non-duplicate ACK is received, TCP Vegas checks for the expiration of the timer again. If so, TCP Vegas retransmits another lost segment but congestion window is not decreased again. This idea is vary similar to fast recovery of New Reno.

2. Innovative Congestion Avoidance: TCP Vegas tries to detect the forthcoming congestion by comparing the current measured throughput to the expected throughput. For each RTT, the sender calculates $Diff$ ($expected\ throughput - measured\ throughput$). Two parameters, α and β ($\alpha < \beta$), are maintained in TCP Vegas. The congestion window is increased linearly in the next RTT if $Diff < \alpha$. If $Diff > \beta$, the congestion window is decreased linearly in the next RTT. If $\alpha < Diff < \beta$, the congestion window will not be changed in the next RTT.

3. Modified Slow Start: Slow Start mechanism is modified to avoid segment loss in SS state. The innovative congestion detection mechanism in CA is also applied in SS state. In order to compare the expected and the actual throughput, the congestion window is allowed to grow only every other RTT.

In fact, we can regard approved TCP implementations as a Reactive TCP which only reacts to the loss of segment. We can also regard TCP Vegas as a Reactive TCP which reacts to the loss of segment and the relationship of measured throughput and expected throughput. In our Reactive TCP, we try to react to more parameters in order to improve TCP performance.

2.2 TCP and Network Path Characteristics

According to above analysis, different algorithms may be used for a function in different TCP implementations. An algorithm may be better than other algorithms over some links and worse over other links. For example, normally, Fast Retransmission can detect congestion more quickly than Timeout. But Fast Retransmission will cause spurious retransmission if the network path frequently transmits packets out of order. Except above algorithms used by Tahoe, Reno, New Reno, and Vegas, many enhancements have been proposed for different link characteristics. So, it is valuable to investigate network path characteristics which affect TCP performance and their corresponding TCP enhancements.

The characteristics of a network path can be represented by Available Bandwidth (ABW), Round Trip Time (RTT), RTT Variance (or jitter), Packet Loss Rate (PLR), Packet Reorder, and Asymmetry. We will discuss their effects to TCP and corresponding enhancements in the following paragraphs.

2.2.1 Available Bandwidth and Round Trip Time

Available Bandwidth (ABW) of a network path is the available bandwidth of its bottleneck link. It is the highest throughput that a TCP connection can achieve. RTT (Round Trip Time) is the sum of propagation delay of all links of a network path and queue & process delay of all routers of the network path. ABW and RTT determine the Bandwidth Delay Product (BDP) of a network path.

$$BDP = ABW * RTT$$

The buffer size of the sender and receiver should be larger than BDP so that TCP throughput will not be constrained by the buffers. TCP Buffer Auto-tuning [59] is proposed to adjust these buffers according to BDP in order to efficiently support large number of connections whose BDP may vary a lot.

The BDP value of a network path also affects TCP protocol in other ways. If BDP is large, fast recovery should be used to avoid abrupt decrease of sending rate when congestion occurs. If BDP is very large, multiple segment loss may occur during one RTT. New Reno [33] or SACK [49] should be used in this case. If BDP is larger than 65535, TCP protocol can not fully utilize bandwidth provided by network. Window Scale Option [41] should be used. It expands the definition of the TCP window to 30 bits through a scale factor. The scale factor is carried in Window Scale Option which is sent only in a SYN segment (a segment with the SYN bit on). The window scale is fixed in each direction after a TCP connection was established.

If BDP is very small, TCP also faces several problems. Firstly, congestion may occur frequently. When TCP sender probes available bandwidth in SS or CA, the sending rate can be larger than ABW quickly and segments are lost. Next, if BDP is very small and a segment is lost, TCP sender can not send enough segments to generate three duplicate

ACKs which trigger fast retransmit and fast recovery. This means that a retransmission timeout is required to recover from the loss. Limited Transmit [12] is proposed to solve this problem. It suggests the sender to send a new segment when the first and second duplicate ACK packets are received. By this way, the receiver is more likely to be able to continue to generate duplicate ACKs and trigger fast retransmission & fast recovery.

RTT is gotten by TCP sender through measuring the time interval between sending a segment and receiving the corresponding acknowledgment. The mean and variance of RTT determine RTO. Thus, it is very important to measure RTT accurately and timely. Normally, TCP sender only measures one RTT sample per window. Time Stamp Option is proposed to almost sample one RTT for each received ACK. Details of RTT Measurement with Time Stamp Option is given in [41]. This mechanism has been used by TCP Vegas.

RTT also affects the response time of a TCP connection. TCP is a self-clocking protocol. The sender increases CWND when ACK is received. If RTT is short, ACK can be fed back quickly and CWND of TCP sender can be opened quickly. Hence, TCP sender can probe available bandwidth quickly.

If RTT is too large, TCP faces several problems. Firstly, TCP sender can not open window quickly due to long RTT. ACK Countering [16] and ACK-every segment in Slow Start propose the receiver to send more ACKs so that TCP sender can open window quickly. Larger Initial Window [13] is proposed to set large value, such as 2, 3, and 4, to initial value of CWND. Secondly, The three-way handshake of TCP connection establishment consumes too much time for short TCP connections, such as HTTP connections. T/TCP [26] propose to exchange data in parallel with the connection establishment in order to reduce response time for users.

Available bandwidth may change due to cross traffic, re-route, network interface change at end points, etc. The congestion control of TCP can handle the changes due to cross traffic well. But classic TCP implementation can not adapt to abrupt changes due to re-route, network interface change, etc. It can not probe increased bandwidth quickly and causes many packets loss when available bandwidth is decreased. Explicit Notify, such as handoff notification [18], may be a solution.

RTT may also change due to many reasons. We will summarize the reason of RTT change, the effect of RTT variance, and proposed mechanisms in the next subsection.

2.2.2 RTT Variance

RTT Variance or jitter is the variance of RTT. RTT may change due to a lot of reasons, such as the change of queue delay, re-route, link layer retransmission, etc.

Large RTT Variance affects TCP performance in several ways. Firstly, large RTT variance causes a large value of RTO. This slows down TCP response speed to congestion. In this case, Fast Retransmission should be used to detect congestion by three duplicate ACKs. Secondly, if RTT changes abruptly, Spurious Timeouts [35] may occur. Due to Spurious Timeout, outstanding segments are retransmitted unnecessarily. These segments will trigger duplicate ACKs at the receiver. Thus, spurious fast retransmission is triggered and results in poor TCP performance. Eifel algorithm [46] uses Time Stamp Option to detect spurious timeouts and eliminates unnecessary retransmission, hence the following spurious fast retransmission.

2.2.3 Packet Reordering

Packet Reordering is not a rare event for TCP. Different segments may use different paths of IP networks. And some routers may reorder packets for optimization. Paxson [53] reports the reordering observed in TCP transfers on a mesh of 35 measurement hosts. This study shows that 0.1%-2.0% 0f all segments (data and ACK) experience reordering in the network. Packet Reordering affects TCP in the following ways.

Firstly, the reordering of TCP segments and ACKs interrupts TCP's self-clock mechanism [42]. Segment reordering triggers that the receiver sends a new ACK (which opens window in a large step) after several duplicate ACKs. Thus, the transmission of TCP sender is more bursty.

Secondly, when packet reordering is larger than three and fast retransmission is used, spurious fast retransmission is triggered. Unnecessary retransmission will waste bandwidth, and unnecessary sending rate deduction due to the following fast recovery worsens TCP performance. DSACK [34] and Eifel algorithm [46] are proposed to detect spurious retransmission. Hence, unnecessary congestion recovery can be avoided. Allman [24] proposes to adjust duplicate ACK threshold in order to avoid spurious fast retransmission in out-of-order networks. In this case, the benefit of fast retransmission is lessened. And if Limit Transmit algorithm [12] is used, the algorithm should be extended to send a segment for Threshold-1 duplicate ACKs.

2.2.4 Packet Loss Rate

Packet Loss Rate (PLR) is the probability that a packet is dropped at any router (congestion) or corrupted on any link (transmission error) of a network path. TCP is designed for high reliable links and regards segment loss as the signal of congestion. TCP can handle infrequently segment loss due to congestion well by its AIMD congestion control. But high PLR caused by transmission error brings serious problems to TCP.

The loss of ACK will cause bursty transmission at TCP sender. The loss of segment due to transmission error violates TCP's assumption that segment is lost due to congestion. The sending rate will be decreased unnecessarily and TCP performance is very poor.

Normally, PLR due to transmission error is low in core network. Access link, especially wireless link, is the main place that a packet is corrupted. A lot of mechanisms have been proposed for lossy wireless links. They can be classified into three categories.

1. End-to-End proposals: End-to-End proposals try to make the TCP endpoints aware of high PLR of the access link. The changes are restricted to the endpoints.

   (a) Fast retransmission and fast recovery [15]: With this mechanism, TCP sender can recover from packet corruption quickly and avoid abrupt decrease of sending rate.

(b) SACK (Selective Acknowledgment TCP option) [49] : SACK option brings selective retransmission into TCP. It is very useful to recover from multiple segment loss in one sending window, which is common over wireless links due to their high PLR. But it may destroy TCP/IP header compression mechanism if it exists.

(c) Distinguishing congestion and corruption: ELN (Explicit Loss Notification) mechanisms, such as HACK [57], which know the packet is lost due to corruption, will inform the peer. When TCP sender knows the reason of segment loss, it can avoid unnecessary congestion recovery.

(d) Use of Small MSS: This mechanism can decrease the probability that a segment is corrupted. But the overhead of TCP/IP header increases.

2. Split-Connection proposals: In these proposals, such as I-TCP [18] and SNOOP [22], TCP connection is divided into two segments. One is between end host and base station of a wireless network, the other is between mobile terminal and base station. The latter can be optimized for wireless link, such as local retransmission. Split-Connection proposals violate the end-to-end semantics of TCP. PEP (Performance Enhancing Proxy) [25] discusses split-connection proposals in details.

3. Link Layer proposals: Since packet is mainly lost over access link, it is reasonable to solve this problem at the link layer. Link Layer proposals use local retransmission at link layer to hide packet loss from TCP. RLP [8] of GSM and RLC [9] of GPRS/UMTS are examples of this category. They both use selective retransmission

and basic flow control schemes. In these proposals, the layer structure of OSI is neatly kept. But it will bring large RTT variance whose effects have been investigated in section 2.2.2. Link layer may also use adaptive coding schemes to fully utilize wireless resources, use Forward Error Correction (FEC) to reduce segment loss rate, and use header compression to decrease header overhead on wireless link.

In Reactive TCP, only end-to-end proposals are considered.

2.2.5 Asymmetry

Asymmetry means that network path characteristics of the two directions are different. It mainly occurs when asymmetric access link, such as ADSL, Satellite and GPRS, is used. As for these links, the bandwidth of uplink is much less than that of downlink. They are designed in this way based on the assumption that data downloaded by users is more than data uploaded to network. If the difference is very large, the large bandwidth of downlink can not be utilized because the reverse direction can not transmit ACK packets generated by the receiver [19]. Large MSS, ACK Congestion Control, WPM [11], ACE [38], and TCP Byte Counting [17] are proposed to solve this problem. Balakrishnan [21] summarized these mechanisms.

In this section, we have discussed the effects of network path characteristics and TCP enhancements for specific links. At the end of this chapter, table 2.1 lists algorithms for TCP receiver functions and their proposed links. Table 2.2 summarizes different algorithms for TCP sender functions.

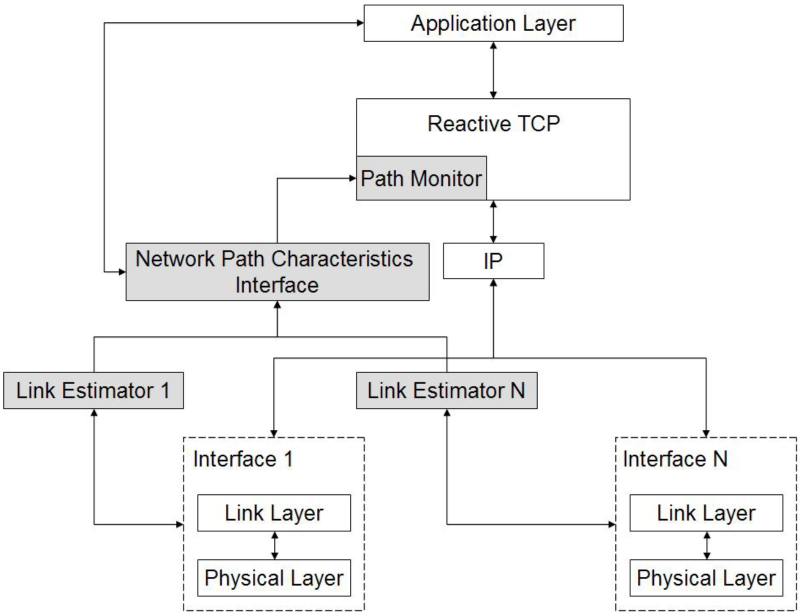

Figure 2.3: Protocol Framework

2.3 Protocol Framework for Multiple Interfaces

Currently, more and more devices are installed with multiple interfaces. With the support of mobile IP, a TCP connection can survive across different interfaces. Reactive TCP need know characteristics of current interface in order to select algorithms. We propose a framework of protocol stack to provide network path characteristics to Reactive TCP and other adaptation protocols. Figure 2.3 shows the protocol stack framework proposed by us.

Network Path Characteristics Interface is the unified interface that provides network path characteristics to Reactive TCP and other adaptation protocols. This interface accepts input from many sources. It may be an application which measures network path characteristics by some algorithms, such as pathload [44], pathchar [43], pathChirp [56], etc. It may also be a Link Estimator of the currently used interface which estimates link characteristics through monitoring communications over the interface. In chapter 3, we propose such a Link Estimator for IEEE 802.11 DCF based WLAN.

The consumers of network path characteristics, adaptation protocols such as Reactive TCP, can register them to Network Path Characteristics Interface so that the interface can push network path characteristics to the consumers. The consumers can also query this interface when it needs to know network path characteristics.

Path Monitor of Reactive TCP gets network path characteristics or link characteristics from the unified interface. If only the characteristics of access link are available, Path monitor can deduce network path characteristics from link characteristics and statistics of TCP, such as RTT, segment loss rate, etc. For example, if RTT is much larger than delay of the access link, the variance of link delay may not cause spurious time out.

Pradeep Sudame [62] also proposed a framework to support adaptation under multiple interfaces. The device with multiple interfaces monitors currently used interfaces and reports the change of interface to upper layers through an ICMP packet which is generated by IP layer of this device. This framework assumes that only lower layers can know network path characteristics. Our proposal has no such constrain. Network path characteristics can be measured by applications and reported to a unified interface.

In this chapter, we analyzed TCP protocol in order to design Adaptive TCP implementation. We also summarized different algorithms proposed for different links. This work is helpful to design reactive rules for Reactive Engine. In the next chapter, we present how to estimate the characteristics of wireless links, the input of Reactive TCP.

Table 2.1: Functions and Algorithms of TCP Receiver

Function	Algorithms	Comments
ACK Content	Std. Algorithm	(the next expected sequence number)
	SACK	lossy links and LFNs
	DSACK	to detect spurious retransmission
ACK Frequency	Std. algorithms in CA	1 ACK/ 2 segment
	Std. algorithms in SS	1 ACK/segment
	2 ACK/segment	slow links
	1 ACK/multiple segments	Asymmetric links
Flow Control	Std. Algorithm	(16-bits Window Field)
	Window Scale Option	(Extended to 30-bits) LFN

Table 2.2: Functions and Algorithms of TCP Sender

Function	Algorithms	Comments
CWND Initial Value	Std. Algorithm	(CWND = 1)
	Larger Initial Window	(CWND=2,3,4,...) slow links
Slow Start	Std. Algorithm	(CWND + 1 per ACK)
	TCP Byte Counting	slow links, asymmetric links
Congestion Detection	RTO	
	3 DUP- ACK	fast retransmission lossy links, fat pipes
	N DUP-ACK	networks with packet reordering
	ELN(HACK)	lossy links
	Using Inter Arrival Time	lossy links
Congestion Avoidance	Std. Algorithm	(CWND+1 per Window) generate congestion and recovery
	Vegas CA	avoid congestion
Congestion Recovery	Slow Start	
	Fast Recovery	avoid abrupt CWND decrease
	New Reno	for multiple segment loss
RTT Measurement	Std. Algorithm	(1 RTT sample/window)
	RTTM with Timestamp Option	(almost 1 RTT sample/segment) used by Eifel, Vegas
Retransmission	Std. Algorithm	go-back-N
	SACK	lossy links
Spurious Timeout Detection	DSACK	
	Eifel	more robust than DSACK

Chapter 3

Wireless Link Characteristics Estimation

Due to the dynamics of wireless link and potential vertical handoff, Reactive TCP should be a solution for mobile nodes. Reactive TCP needs the link characteristics so that it can select appropriate algorithms according to current wireless link characteristics.

Many algorithms have been proposed to estimate network path characteristics (especially ABW), such as Delphi [55], pathload [44], pathchar [43], and [56]. They send probing-packets and deduce network path characteristics by analyzing delay experienced by these probing-packets.

These intrusive algorithms are not appropriate when a wireless link is used in a network path. Firstly, probing-packets consume precious bandwidth of a wireless link. Next, the characteristics of a wireless link change frequently and abruptly due to mobility, fading and contention among nodes. Intrusive algorithms could not measure network path characteristics accurately and timely with small cost (bandwidth consumed by probing-packets). Thus, non-intrusive algorithms, which estimate wireless link characteristics at the end point (mobile node), are valuable for Reactive TCP.

In this chapter, we propose a new non-intrusive mechanism to estimate link characteristics of IEEE 802.11 DCF based WLAN, one of the most popular wireless access networks. The link characteristics of a WLAN are affected by characteristics of a wireless channel and contention among mobile nodes. It is a good starting point to design link characteristics estimation algorithms for wireless links.

This chapter comprises three sections. In section 1, we introduce IEEE 802.11 DCF based WLAN. We highlight several related works in section 2. In section 3, we present our link characteristics estimation mechanism for IEEE 802.11 DCF based WLAN.

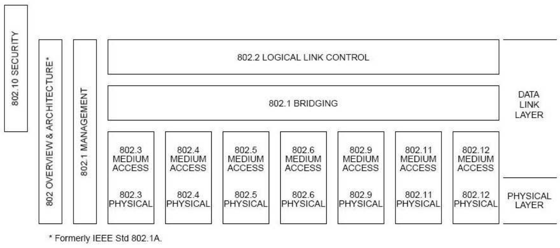

Figure 3.1: IEEE Standards for LAN & MAN (from [5])

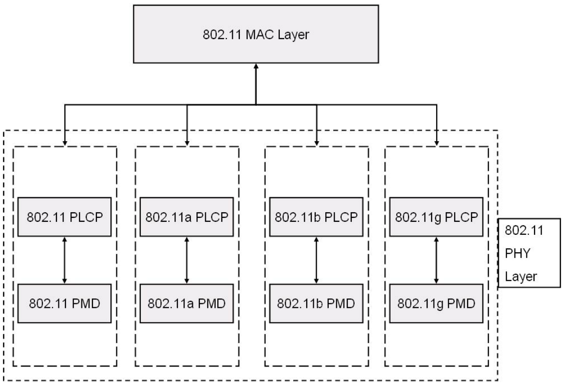

Figure 3.2: IEEE Standard for Wireless LAN

3.1 IEEE802.11 DCF Based WLAN

IEEE 802.11 standard was approved by LAN MAN Standards Committee of the IEEE Computer Society for LAN over wireless medium. It is part of a family of standards for local and metropolitan area networks. Figure 3.1 depicts the whole standard family.

IEEE 802.11 standard includes a series of specifications which standardize MAC and physical layers of WLAN. This standard includes several different physical layers (802.11 [5], 802.11b [7], 802.11a [6], and 802.11g) which use the same MAC protocol. Figure 3.2 gives an overview of specifications of IEEE 802.11 standard.

Table 3.1: PMDs of IEEE 802.11

Standard	802.11	802.11b	802.11a	802.11g
PMD	FHSS, DSSS,IR	DSSS	OFDM	OFDM
Frequency (GHz)	FHSS: 2.4-2.497 DSSS: 2.4-2.497 IR: Infrared	2.4-2.497	5.15-5.35 5.425-5.675 5.725-5.875	2.4-2.497
Data Rate (Mbps)	DSSS: 1, 2 FHSS: 0.5-4.5 IR: 1, 2	CCK: 1, 2, 5.5, 11	OFDM: 6, 9, 12, 18 24, 36, 48, 54	OFDM: 6,...,54 CCK: 1,2,5.5,11
Channel	4	4	8	4

Each physical layer includes two sub-layers, physical layer convergence procedure (PLCP) and physical medium dependent (PMD). PLCP is a convergence procedure to map PDU from MAC into a frame whose format is designed for radio transceiver of corresponding PMD which provides the actual means to transmit data on medium. Table 3.1 summarizes the techniques of these different PMDs. IR is seldomly used in practice and the deployment based on original 802.11 is being substituted by 802.11b, 802.11a, and 802.11g. WLANs based on 802.11b dominate the currently WLAN deployment.

Frame is the transmission unit of IEEE 802.11 WLAN. A frame includes preamble of PMD, PLCP header, MAC header, and potential data from upper layers. These parts may be sent with different modulation schemes. Figure 3.3 (next page) shows the format of a frame in IEEE 802.11b WLAN.

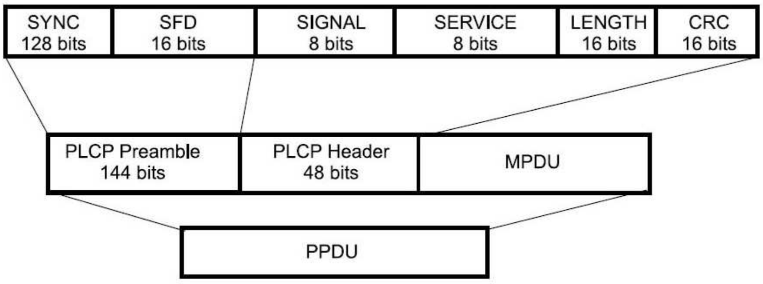

Figure 3.3: IEEE 802.11 Frame Structure (From [5] Fig.86)

Table 3.2: Difference between Long Preamble and Short Preamble

	Long Preamble	Short Preamble
Preamble Length (bit)	144	72
Preamble Data Rate (Mbps)	1	1
PLCP Header Length (bit)	48	48
PLCP Header Data Rate (Mbps)	1	2

In order to reduce overhead of physical layer, short preamble is introduced in IEEE 802.11b. Table 3.2 shows the difference in frame header between long preamble and short preamble.

IEEE 802.11 standard supports two network types: ad hoc network and infrastructure network. Ad hoc network is a hot research field, but most of IEEE 802.11 enabled nodes are used in infrastructure network mode. These nodes use IEEE 802.11 based WLANs to access Intranet and/or Internet. IEEE 802.11 based WLAN perhaps is the most popular wireless access network. More and more mobile devices, such as laptop and PDA, have been installed with IEEE 802.11 interface.

IEEE 802.11 standard supports two different media access control functions: DCF (Distributed Coordination Function) and PCF (Point Coordination Function). DCF is known as carrier sense multiple access with contention avoidance (CSMA/CA). PCF is a contention free access method which is designed for infrastructure mode. But most of current IEEE 802.11 based WLANs use DCF because of its simplicity [60]. In this thesis, we focus on IEEE802.11 DCF based WLAN. In the following parts of this chapter, WLAN refers to IEEE802.11 DCF based WLAN.

3.1.1 Distributed Coordination Function

In DCF of WLAN, basic access method is the core mechanism that a node uses to determine whether it may transmit data. RTS/CTS may be used to solve hidden node problem [23].

In basic access method, whenever a node is ready to send a packet, it generates a random back-off timer chosen uniformly from $[0, W - 1]$, where W is the contention window. Initially, W is set to CW_{min}. Then the node senses the channel to be idle for a period of Distributed Inter Frame Spacing (T_{DIFS}). After that, the back-off timer begins to decrease. During that, the timer may pause if other nodes begin to transmit over the channel. If so, the timer will resume when the channel is idle for T_{DIFS} again. When the back-off timer expires, the node begins to send the packet within a DATA frame under the assumption that fragmentation is not used in MAC layer. When the receiver gets the DATA frame correctly, the receiver waits for a period of Short Inter Frame Spacing (T_{SIFS}) and sends back an ACK frame. If the sender receives the ACK frame correctly, the packet is transmitted successfully. If collision occurs or DATA/ACK frame is corrupted, the sender doubles its contention window (but no larger than CW_{max}), sets a new back-off timer, and tries to

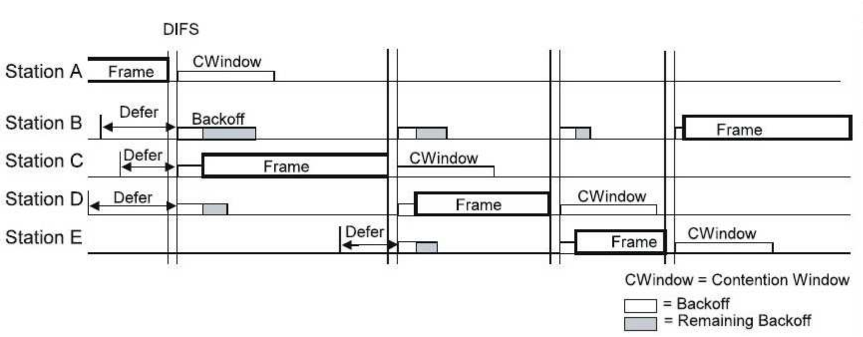

Figure 3.4: Back-off Procedure (From [5] Fig.52)

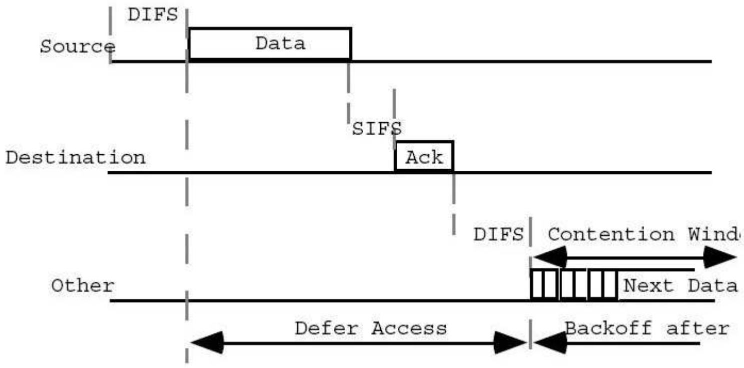

Figure 3.5: Frame Sequence of Basic Access Method

transmit again until the packet is transmitted successfully or discarded after a threshold of retransmission time ($RETRY$). Figure 3.4 depicts the back-off procedure of several nodes which are contending a WLAN channel.

In RTS/CTS access method, instead of exchanging DATA/ACK frames directly, the sender and the receiver first exchange RTS/CTS to reserve the whole channel. After that, the DATA/ACK frames will be exchanged. Figure 3.5 and 3.6 depict the frame exchange sequence of basic access method and RTS/CTS access method.

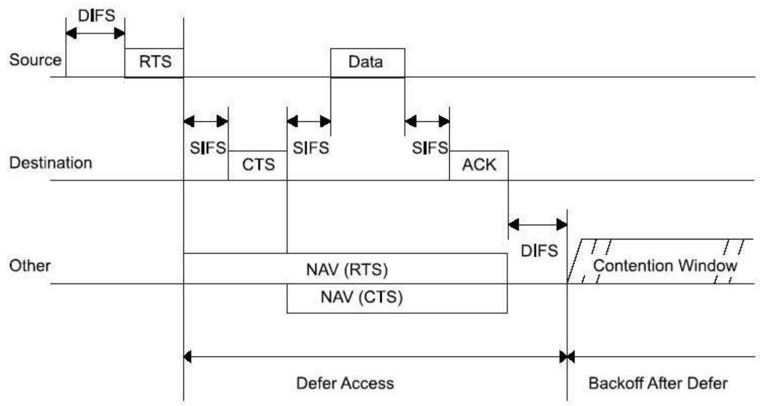

Figure 3.6: Frame Sequence of RTS/CTS Access Method (From [5] Fig.53)

3.1.2 WLAN Channel

The characteristic of a WLAN channel is that of a typical wireless channel. BER is determined by wireless link quality, modulation, and code schemes [45].

Wireless link quality can be represented by Signal to Noise Ratio (SNR) of frames at the receiver. Signal means the frame signal strength measured at the receiver. Noise includes noise generated by the receiver, noise from environment, and interference caused by other frames which is received simultaneously. Noise generated by the receiver includes thermal noise and platform noise. Different products may generate different noise due to different platform noises. And the noise for bits transmitted with different data rates is different due to the difference of their thermal noises. Since different data rates are used within a frame, SNRs experienced by different parts of a frame are also different.

Since IEEE 802.11 has no error correction code, BER of IEEE 802.11 DCF based WLAN

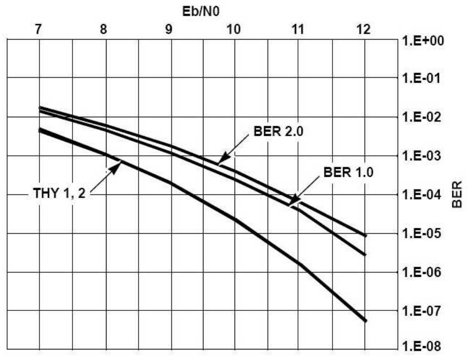

Figure 3.7: BER vs Eb/N0 Performance for PSK Modes

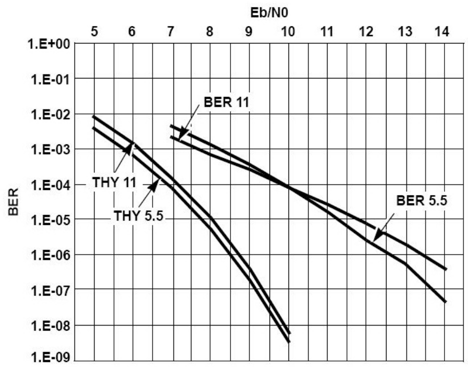

Figure 3.8: BER vs Eb/N0 Performance for CCK Modes

channel is determined by SNR and modulation scheme or transmission rate.

$$BER = ber(SNR, Rate)$$

The curves of BER*vs.*SNR can be derived theoretically or measured with real products. For example, Sklar [61] has derived the relationship of BER and SNR under QPSK modulation scheme. Intersil has provided the theoretical and measured curves for HFA3861B Chipset [37] in Figure 3.7 and 3.8. To be close to the reality, we use these curves measured with HFA3861B Chipset.

In IEEE 802.11 DCF based WLAN, the transmission rates of preamble and PLCP header are constant. And the transmission rate of a MAC frame will not change within the same frame. Due to the short transmission duration of a frame and the low-speed movement supported by WLAN, the SNRs will not change much during the transmission of a frame [58]. So, the Frame Error Rate (FER) due to wireless transmission error is,

$$\begin{aligned} FER &= 1-(1-ber(SNR_{pre},R_{pre}))^{L_{pre}} * (1-ber(SNR_{plcp},R_{plcp}))^{L_{plcp}} \\ &\quad * (1-ber(SNR_{mac},R_{mac}))^{8L_{mac}} \end{aligned} \tag{3.1}$$

Note: R_{pre}—*transmission rate of preamble*

R_{plcp}—*transmission rate of PLCP header*

R_{mac}—*transmission rate of MAC frame*

L_{pre}—*length of preamble in bits*

L_{plcp}—*length of PLCP header in bits*

L_{mac}—*length of MAC frame in bytes*

SNR_{pre}—*SNR experienced by preamble*

SNR_{plcp}—*SNR experienced by PLCP header*

SNR_{mac}—*SNR experienced by MAC frame*

In basic access method, a DATA frame and an ACK frame are needed to transmit a packet from upper layers under the assumption that the packet need not be fragmented. If Data/ACK frame is corrupted or collision occurs, retransmission is needed. The probability that the frame exchange sequence fails is,

$$FSER_b = 1-(1-FLR_{data}) * (1-FER_{ack}) \tag{3.2}$$

$$FLR_{data} = 1-(1-FER_{data}) * (1-P_c)$$

Note: P_c *is the collision probability determined by the whole channel, and the transmission rates of Data frame and ACK frame may be different.*

As for RTS/CTS access method,

$$\begin{aligned} FSER_r &= 1-(1-FLR_{rts})*(1-FER_{cts}) \\ &\quad *(1-FER_{data})*(1-FER_{ack}) \\ FLR_{rts} &= 1-(1-FER_{rts})*(1-P_c) \end{aligned} \tag{3.3}$$

Normally, a DATA frame is much longer than control frames, and higher data rate is used for DATA frame. The probability that a frame exchange sequence fails is dominated by the DATA frame. In IEEE802.11 DCF based WLAN, the sender can not know the SNR experienced by a DATA frame at the receiver. If the wireless environment of the sender and receiver are similar, the sender can deduce the SNR from SNR experienced by the ACK frame at the sender. To be accurate, we let the receiver feed back SNR experienced by DATA frame to the sender in corresonding ACK frame.

3.1.3 Overhead of MAC/PHY Layers

In order to send a packet from upper layers, the headers of MAC and PHY layers should be added, control frames need to be transmitted over the channel, and back-off procedure also wastes some time. In this sub-section, we analyze the time needed to transmit a packet which need not be fragmented.

In IEEE 802.11 DCF based WLAN, the time needed to transmit a frame is,

$$T_{frm} = T_{prop} + \frac{L_{pre}}{R_{pre}} + \frac{L_{plcp}}{R_{plcp}} + \frac{8L_{mac}}{R_{mac}}$$

For basic access method, without retransmission, the time needed to completely transmit a packet is,

$$T_b = T_{DIFS} + T_{bf} + T_{SIFS} + T_{data} + T_{ack} \tag{3.4}$$

As for RTS/CTS access method,

$$T_r = T_{DIFS} + T_{bf} + 3T_{SIFS} + T_{rts} + T_{cts} + T_{data} + T_{ack} \tag{3.5}$$

Note: T_{prop} *is the propagation time of the channel.* T_{rts} *is the time consumed by RTS frame.* T_{cts} *is the time consumed by CTS frame.* T_{data} *is the time consumed by DATA frame.* T_{ack} *is the time consumed by ACK frame.* T_{bf} *is the time consumed by back-off timer before frames start to be transmitting. It is determined by the whole channel.*

3.2 Related Works

In order to estimate WLAN link characteristics experienced by a mobile node, the wireless link quality and its communication protocols must be considered. Wireless link quality determines BER of the wireless link. PHY and MAC protocols determine the protocol overhead. MAC protocol also determines the contention among mobile nodes of a WLAN. The related works in this area are discussed below.

Zhang [65] proposes a non-intrusive link bandwidth estimation model which estimates link bandwidth based on wireless link quality—SNR. BPNN (Back-Propagation Neural Network) and Bayesian Inference methods are used in this model. Measured SNR and link bandwidth are used to train this model. After that, the model predicts link bandwidth according to current link quality. This model does not consider contention among mobile nodes. It focuses on bandwidth estimation of a point-to-point wireless link, and needs a lot of training data to achieve some level of accuracy. This model can not be directly used to estimate link characteristics by mobile nodes of a WLAN. But it points out the effects of wireless link quality (SNR) and the direction to use non-intrusive methods to estimate wireless link characteristics.

Tay [64] establishes a model to estimate the throughput of a WLAN channel. But only basic access mode over an error-free channel is considered. And it assumes that every node is a saturated node which always has data to be sent. It focuses on the effect of contention among saturated nodes and gives the relationship of contention window, collision probability, and the number of saturated nodes. Below is the relationship given by Tay.

$$
\begin{aligned}
\overline{CW} &= \frac{1 - P_c - P_c * (2P_c)^{RETRY}}{1 - 2P_c} * \frac{CW_{min}}{2} \\
P_c &= 1 - (1 - \frac{1}{\overline{\overline{CW}}})^{N-1} \Longrightarrow \\
P_c &= 1 - (1 - \frac{2 * (1 - 2P_c)}{1 - P_c - P_c * (2P_c)^{RETRY}} * \frac{1}{CW_{min}})^{N-1}
\end{aligned}
$$

According to the back-off procedure of IEEE 802.11 DCF, when multiple senders contend a WLAN channel, the probability of a mobile sender to acquire the channel is in inverse proportion to its average contention window. This fact is very important to analysis how mobile nodes share the bandwidth of a WLAN channel. The EDCF of IEEE 802.11e has utilized this fact to support priority among data flows.

These works provide a solid background to estimate WLAN link characteristics for mobile nodes. In the following section, we propose a new mechanism to estimate link characteristics experienced by a mobile node while contending a lossy WLAN channel with other nodes that are saturated or not.

3.3 WLAN Link Characteristics Estimation Mechanism

In our WLAN link characteristics estimation mechanism, the following metrics of a WLAN link are estimated.

- ABW: available bandwidth of the link
- PLR: packet loss rate of the link
- $\overline{LTT}$: average time needed to transmit a packet between mobile node and AP
- LTT_{var}: variation of time needed to transmit a packet between mobile node and AP

Wireless link quality, overhead of MAC & PHY protocols, and contention among nodes determine the link characteristics experienced by a mobile node. The effects of wireless link quality and protocol overhead have been analyzed in section 3.1.2 and section 3.1.3. As for contention, it causes collision and determines how the bandwidth is shared among senders. Normally, there are many mobile nodes in a WLAN channel. These nodes and AP contend the channel to transmit data according to IEEE 802.11 DCF. In this section, we shall analyze the effect of contention and present our link characteristics estimation mechanism.

In our model, frames may be lost due to collision and wireless transmission error. The senders may be saturated senders or not, and the unsaturated senders send data randomly. So, the relationship of average contention window, collision probability, and the number of saturated nodes, which is given by Tay [64], should be changed into the following equations.

$$\overline{CW} = \frac{1 - \overline{FSER} * (1 + (2\overline{FSER})^{RETRY})}{1 - 2\overline{FSER}} * \frac{CW_{min}}{2} \tag{3.6}$$

$$P_c = 1 - (1 - \frac{1}{\overline{CW}})^{M-1} \tag{3.7}$$

$$M = \frac{N_{all}}{N_{s_{max}}} \tag{3.8}$$

$$T_{bf} = \frac{\overline{CW}}{M} \tag{3.9}$$

Note: $\overline{CW}$ *is the average contention window for all senders.* $\overline{FSER}$ *is the average probability for the whole channel that a frame exchange sequence fails.* P_c *is the average collision probability for all senders.* M *is the number of effective saturated nodes.* N_{all} *is the number of frame exchange sequences transmitted over the channel.* $N_{s_{max}}$ *is the number of frame exchange sequences transmitted by the node which gets the largest probability to transmit. Normally, this node is a saturated node. If the channel is idle for a large amount of time, we can assume that* M *is less than one and* P_c *is zero.*

If we can monitor all communications of the whole channel, N_{all} and $N_{s_{max}}$ could be calculated easily. With the number of DATA/ACK/RTS/CTS frames and the number of collisions of the channel, $\overline{FSER}$ can be calculated by Equation 3.10. Thus, $\overline{CW}$, P_c, M, and T_{bf} can be deduced by Equation 3.6, 3.7, 3.8, and 3.9.

$$\overline{FSER} = \frac{(N_{data} - N_{ack}) + (N_{rts} - N_{cts}) + 2N_c}{N_{data} + N_{rts} + 2N_c} \tag{3.10}$$

When collision occurs, the channel is wasted. The cost of collision is the time spent by the longest frame of the collided frames. The average cost of a collision is,

$$C_c = (P_{d+d} + P_{d+r}) * (T_{DIFS} + T_{bf} + \overline{T_{b\_data}}) + P_{r+r} * (T_{DIFS} + T_{bf} + T_{rts}) \tag{3.11}$$

Note: *P_{d+d} is the probability that DATA frame collides with DATA frame, and so on. They can be calculated with knowledge of N_{b_data} (the number of data frames transmitted with basic access method) and N_{rts} for the whole channel. Note that $\overline{T_{b\_data}}$ is the average time spent by a DATA frame transmitted with basic access method.*

According to above analysis, in order to deduce parameters which affect link characteristics experienced by mobile nodes, we should know all communications over the WLAN channel. AP is the proper location to monitor communications, deduce these parameters, and broadcast them to all mobile nodes periodically. After that, a mobile node could use these parameters and the quality of wireless link between AP and the node to estimate link characteristics. Figure 3.9 (next page) depicts this mechanism.

In the following sub-sections, we first present communications monitored by AP and a mobile node. We then give the algorithms used to estimate link characteristics experienced by a sender and a receiver.

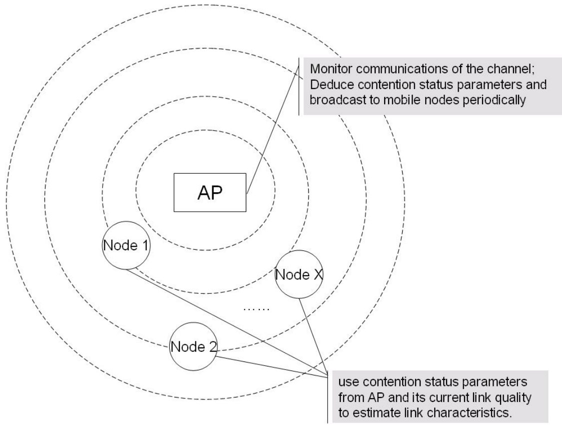

Figure 3.9: Proposed WLAN Link Estimation Mechanism

3.3.1 AP

In this mechanism, AP records the following information for each sender and receiver (a mobile node may be regarded as both sender and receiver, but AP can only be regarded as a sender):

- N_b: the number of frame exchange sequences transmitted with basic access method
- L_b: average length of packets transmitted with basic access method
- N_r: the number of frame exchange sequences transmitted with RTS/CTS access method
- L_r: average length of packets transmitted with RTS/CTS access method
- SNR: link quality between AP and the mobile sender or receiver. For the sender,SNR is measured by AP and fed back to the sender. For the receiver, SNR is measured by mobile nodes and fed back to AP.

Except that, in order to calculate $\overline{FSER}$, AP should record the number of DATA, ACK, RTS, and CTS frames sent by all senders. N_c (collision number), and T_{idle} (idle time of the whole channel) should also be recorded.

Periodically, AP calculates the following parameters of the whole channel based on communications of the last period:

- P_c: average collision probability calculated by Equation 3.7.
- C_c: average collision cost calculated by Equation 3.11.
- T_{bf}: time wasted by back-off timer. It is calculated by Equation 3.9.
- T_{idle}: idle time of the WLAN channel during the last period.
- $\overline{T_{FS}}$: average time needed by one frame exchange sequence. AP first calculates T_b (Equation 3.4) and T_r (Equation 3.5) for every sender. Next, it calculates,

  $$T_s = \frac{N_b * T_b + N_r * T_r}{N_b + N_r}$$

  Finally, AP can calculate the average value of all senders.

  $$\overline{T_{FS}} = \frac{\sum_{i=1}^{Num\_sender}(T_{s_i} * N_{s_i})}{\sum_{i=1}^{Num\_sender} N_{s_i}}, where \ \ N_{s_i} = N_{b_i} + N_{r_i}$$

- N_{all}: the number of all frame exchange sequences transmitted over the channel.

  $$N_{all} = \sum_{i=1}^{Num\_sender} N_{s_i}$$

- $N_{s_{max}}$: the number of frame exchange sequences sent by the sender which gets the largest opportunity to transmit.

  $$N_{s_{max}} = MAX\{N_{s_i}\}, \ \ where \ \ N_{s_i} = N_{b_i} + N_{r_i}$$

- $CW_{s_{max}}$: average contention window of the sender which gets the largest opportunity to transmit. It is calculated by Equation 3.6 with $FSER_{s_{max}}$. $FSER_{s_{max}}$ is calculated from $FSER_b$ (Equation 3.2) and $FSER_r$ (Equation 3.3) of the sender.

$$FSER_{s_{max}} = \frac{N_b * FSER_b + N_r * FSER_r}{N_b + N_r}$$

- $N_{ap}, T_{ap}, FSER_{ap}$: N_{ap} is the number of frame exchange sequences sent by AP. T_{ap} is the average time spent by the frame exchange sequences sent by AP. $FSER_{ap}$ is the probability that the frame exchange sequence sent by AP fails.

- $N_{recv_{max}}$: the number of frame exchange sequences received by the receiver which gets the largest opportunity to receive from AP.

$$N_{recv_{max}} = MAX\{N_{recv_i}\}, \;\; where \;\; N_{recv_i} = N_{b_i} + N_{r_i}$$

- $FSER_{recv_{max}}$: the probability that frame exchange sequence received by the receiver, which gets the largest opportunity to receive from AP, fails. With link quality fed back from the receiver, it is calculated by similar method used for $FSER_{s_{max}}$.

Since the probability to acquire WLAN channel is in inverse proportion to the average contention window, $N_{s_{max}}$ and $CW_{s_{max}}$ are calculated by AP and broadcasted to mobile nodes that use $N_{s_{max}}$ and $CW_{s_{max}}$ to calculate their chances of acquiring the channel.

Assuming that AP uses round robin algorithm to schedule receiving flows, the probability that a receiver receives data from AP is in proportion to $FSER$. So, $N_{recv_{max}}$ and $FSER_{recv_{max}}$ are also sent to mobile nodes.

N_{ap}, T_{ap}, and $FSER_{ap}$ give the status of AP as a sender. They are used to estimate link characteristics for receivers.

After calculating all these parameters, AP broadcasts them to mobile nodes in a Channel Status frame.

3.3.2 Mobile Node

A mobile node has to monitor two flows: the data sent by it (ss) and the data received from AP (sr). For each flow, N_b, L_b, N_r, and L_r are recorded. Mobile node also must record SNR_s(fed back by AP) and SNR_r(measured by it). SNR_s is used to estimate link characteristics experienced by the sender. SNR_r is used to estimate link characteristics experienced by the receiver.

When mobile node receives the Channel Status frame, it records these parameters from AP. With T_{bf} from AP, it calculates T_{ss} and T_{sr} based on communications of the node between this Channel Status frame and the previous one.

$$T_{ss} = \frac{N_{ss\_b} * T_{ss\_b} + N_{ss\_r} * T_{ss\_r}}{N_{ss}}, \quad where \;\; N_{ss} = N_{ss\_b} + N_{ss\_r}$$

$$T_{sr} = \frac{N_{sr\_b} * T_{sr\_b} + N_{sr\_r} * T_{sr\_r}}{N_{sr}}, \quad where \;\; N_{sr} = N_{sr\_b} + N_{sr\_r}$$

If the mobile sender has not sent any data, N_{ss} is set to 0. And the length of packet and the ratio of packets sent with basic access method to packets sent with RTS/CTS access method may use default values or use input from upper layers. If the mobile receiver has not received any data, the same methods are used.

Based on above information and contention status from AP, mobile sender or receiver first estimates the time spent by other flows between two consecutive transmissions for this flow. After that, mobile node can calculate characteristics experienced by this flow according to wireless link quality and collision probability. The details of these algorithms are given in the following sub-sections.

3.3.3 Algorithms for a Mobile Sender

When a mobile node acts as a sender, it first calculates average time used by a frame exchange sequence of other senders according to the following equation.

$$\overline{T_{FS\_other\_senders}} = \frac{\overline{T_{FS}} * N_{all} - T_{ss} * N_{ss}}{N_{all} - N_{ss}}$$

Next, mobile sender calculates its $FSER_{ss}$ by the following equation.

$$FSER_{ss} = \frac{N_{ss\_b} * FSER_{ss\_b} + N_{ss\_r} * FSER_{ss\_r}}{N_{ss}}$$

$FSER_{ss\_b}$ and $FSER_{ss\_r}$ are calculated according to Equation 3.2 and Equation 3.3. With $FSER_{ss}$, mobile node can calculate CW_{ss} according to Equation 3.6.

If CW_{ss} is much less than T_{bf}, the mobile node should adjust $\overline{T_{FS}}, T_{ap}, T_{ss}$ and T_{sr} by substituting T_{bf} with CW_{ss}. If several nodes with very bad link quality start up firstly, this situation may occur when a node with excellent link quality is powered on. In this situation, CW_{ss} will determine T_{bf} of the whole channel.

When calculating ABW, a node assumes that idle time of the channel will be acquired by itself. Thus, N_{ss} and N_{all} should be added by T_{idle}/T_{ss} before calculating N_{ss_expect} (the expected times that this sender acquires the channel) according to Equation 3.12.

$$N_{ss\_expect} = \frac{N_{s_{max}} * CW_{s_{max}}}{CW_{ss}} \tag{3.12}$$

If N_{ss_expect} is larger than N_{ss}, we should reduce $N_{other\_senders}(N_{all} - N_{ss})$ so that the sender can acquire N_{ss_expect} chance to transmit. After that, if N_{ss_expect} is larger than all other senders and CW_{ss} is not much less than $CW_{s_{max}}$, we should adjust N_{ss_expect} so that all saturated nodes get proper opportunity to transmit.

With N_{ss_expect}, we can calculate the time spent by other senders between two consecutive transmissions of this sender.

$$T_{ss\_other} = \frac{N_{other\_senders} * \overline{T_{FS\_other\_senders}}}{N_{ss\_expect}} \tag{3.13}$$

After T_{ss_other} is calculated, the sender first calculates link characteristics experienced by data frames sent with basic access method. The packet loss rate is,

$$PLR_{ss\_b} = (FSER_{ss\_b})^{RETRY} \tag{3.14}$$

With $C_{ss\_fail\_b}$ (average time wasted by one failed frame exchange sequence of basic access method) calculated by Equation 3.15, we can calculate LTT_i (the time spent by a packet if it is transmitted successfully at the i^{th} transmission) by Equation 3.16.

$$\begin{aligned} C_{ss\_fail\_b} &= \frac{FLR_{data} * C_{data\_loss}}{FSER_{ss\_b}} \\ &\quad + \frac{(1 - FLR_{data}) * FER_{ack} * T_{ss\_b}}{FSER_{ss\_b}} \end{aligned} \tag{3.15}$$

$$
\begin{aligned}
C_{data\_loss} &= \frac{FER_{data} * C_{data\_err}}{FLR_{data}} \\
&+ \frac{P_c * MAX\{C_{data\_err}, C_c\}}{FLR_{data}} \\
C_{data\_err} &= T_{DIFS} + T_{bf} + T_{data} \\
LTT_i &= \sum_{j=1}^{i-1} (2^{j-1} * T_{ss\_other} + C_{ss\_fail\_b}) + T_{ss\_b} \qquad (3.16)
\end{aligned}
$$

With LTT_i and P_i (Equation 3.17), we can calculate $\overline{LTT_{ss\_b}}$ (Equation 3.18), $LTT_{ss\_var\_b}$ (Equation 3.19), and ABW_{ss_b} (Equation 3.20).

$$
\begin{aligned}
P_i &= (1 - FSER_{ss\_b}) * (FSER_{ss\_b})^{i-1} \qquad (3.17) \\
\overline{LTT_{ss\_b}} &= \sum_{i=1}^{RETRY} (P_i * LTT_i) \qquad (3.18) \\
LTT_{ss\_var\_b} &= \sum_{i=1}^{RETRY} P_i * (|\, LTT_i - \overline{LTT_{ss\_b}}\, |) \qquad (3.19) \\
ABW_{ss\_b} &= \frac{8L_{ss\_b}}{\overline{LTT_{ss\_b}}} \qquad (3.20)
\end{aligned}
$$

Next, mobile sender estimates link characteristics experienced by data frames sent with RTS/CTS access method by the following equations.

$$
PLR_{ss\_r} = (FSER_{ss\_r})^{RETRY} \qquad (3.21)
$$

The average time wasted by a failed frame exchange sequence sent with RTS/CTS access method can be calculated by following equations.

$$
\begin{aligned}
C_{ss\_fail\_r} &= \frac{FLR_{rts} * C_{rts\_loss}}{FSER_{ss\_r}} + \frac{(1 - FLR_{rts}) * FER_{cts} * C_{cts\_err}}{FSER_{ss\_r}} \\
&+ (1 - FLR_{rts}) * (1 - FER_{cts}) * \frac{FER_{data} * C_{data\_err}}{FSER_{ss\_r}} \\
&+ (1 - FLR_{rts}) * (1 - FER_{cts}) * (1 - FER_{data}) * \frac{FER_{ack} * T_{ss\_r}}{FSER_{ss\_r}} \\
C_{rts\_loss} &= \frac{FER_{rts} * C_{rts\_err}}{FLR_{rts}} + \frac{Pc * C_c}{FLR_{rts}}
\end{aligned}
$$

$$
\begin{aligned}
C_{rts\_err} &= T_{DIFS} + T_{bf} + T_{rts} \\
C_{cts\_err} &= C_{rts\_loss} + T_{SIFS} + T_{cts} \\
C_{data\_err} &= C_{cts\_err} + T_{SIFS} + T_{data}
\end{aligned}
$$

With similar algorithms used for basic access method, $\overline{LTT_{ss\_r}}$, $LTT_{ss\_var\_r}$, and ABW_{ss_r} can be calculated by the following equations.

$$
\overline{LTT_{ss\_r}} = \sum_{i=1}^{RETRY} (P_i * LTT_i) \tag{3.22}
$$

$$
\begin{aligned}
LTT_i &= \sum_{j=1}^{i-1} (2^{j-1} * T_{ss\_other} + C_{ss\_fail\_r}) + T_{ss\_r} \\
P_i &= (1 - FSER_{ss\_r}) * (FSER_{ss\_r})^{i-1}
\end{aligned}
$$

$$
LTT_{ss\_var\_r} = \sum_{i=1}^{RETRY} P_i * (| LTT_i - \overline{LTT_{ss\_r}} |) \tag{3.23}
$$

$$
ABW_{ss\_r} = \frac{8 * L_{ss\_r}}{\overline{LTT_{ss\_r}}} \tag{3.24}
$$

Finally, the link characteristics experienced by the mobile sender can be calculated.

$$
\begin{aligned}
PLR_{ss} &= \frac{N_{ss\_b} * PLR_{ss\_b} + N_{ss\_r} * PLR_{ss\_r}}{N_{ss\_b} + N_{ss\_r}} \\
\overline{LTT_{ss}} &= \frac{N_{ss\_b} * \overline{LTT_{ss\_b}} + N_{ss\_r} * \overline{LTT_{ss\_r}}}{N_{ss\_b} + N_{ss\_r}} \\
LTT_{ss\_var} &= \frac{N_{ss\_b} * LTT_{ss\_var\_b} + N_{ss\_r} * LTT_{ss\_var\_r}}{N_{ss\_b} + N_{ss\_b}} \\
ABW_{ss} &= \frac{N_{ss\_b} * \overline{LTT_{ss\_b}} * ABW_{ss\_b}}{N_{ss\_b} * \overline{LTT_{ss\_b}} + N_{ss\_r} * \overline{LTT_{ss\_r}}} \\
&\quad + \frac{N_{ss\_r} * \overline{LTT_{ss\_r}} * ABW_{ss\_r}}{N_{ss\_b} * \overline{LTT_{ss\_b}} + N_{ss\_r} * \overline{LTT_{ss\_r}}}
\end{aligned}
$$

3.3.4 Algorithms for a Mobile Receiver

When a mobile node acts as a receiver, the mobile receiver first calculates average time spent by a frame exchange sequence of other receivers.

$$\overline{T_{other\_receivers}} = \frac{T_{ap} * N_{ap} - T_{sr} * N_{sr}}{N_{ap} - N_{sr}}$$

Secondly, the receiver calculates T_{ap_other} for AP with the same algorithm used by mobile sender to calculate T_{ss_other}. One exception is that T_{idle}/T_{sr} should be added to N_{all}, N_{ap}, and N_{sr}. Thirdly, it calculates $FSER_{sr}$ by similar method used for $FSER_{ss}$.

With round robin schedule algorithm, the probability of flows to receive data is in proportion to $FSER$. Thus,

$$N_{sr\_expect} = \frac{N_{recv_{max}} * (1 + FSER_{sr})}{1 + FSER_{recv_{max}}}$$

If N_{sr_expect} is larger than N_{sr}, we should reduce N_{recv_other} $(N_{ap} - N_{sr})$ so that the receiver can acquire N_{sr_expect} opportunity to receive data. After that, if N_{sr_expect} is larger than all other receivers and $FSER_{sr}$ is not much larger than $FSER_{recv_{max}}$, we should adjust N_{sr_expect} so that saturated receivers get proper opportunity to receive data.

After that, the time consumed by other senders and receivers between two consecutive transmissions of this receiver is,

$$T_{sr\_other} \quad = \quad \frac{N_{ap\_expect} - N_{sr\_expect}}{N_{sr\_expect}} * (T_{ap\_other} + \overline{T_{other\_receivers}})$$

Finally, with T_{sr_other}, link characteristics experienced by the mobile receiver can be calculated by the same algorithm used for mobile sender.

In this chapter, we presented our link estimator for IEEE 802.11 DCF based WLAN. In next chapter, we will present how to simulate it in NS2. We also test its accuracy by simulation experiments in chapter 5.

Chapter 4

Simulation Setup

NS2 [2] is the most popular network simulator among the research community. In this chapter, we present how to simulate WLAN link characteristics estimation mechanism, which is proposed in chapter 3, in NS2.26 (the latest version of NS2). We first describe how to simulate a IEEE 802.11b based WLAN channel in NS2. We then present how to implement our estimation mechanism in NS2. In the next chapter, we describe simulation experiments used to test the accuracy of our mechanism.

4.1 WLAN Channel Simulation

NS2.26 uses thresholds to determine whether a frame is received correctly by the receiver [63]. It does not support short preamble. We need change NS2.26 in order to simulate a WLAN channel. We first implement the wireless transmission error model of section 3.1.2 in NS2. We then add support for short preamble. Finally, we investigate parameters to simulate an IEEE802.11b channel in office, the common environment of WLAN.

4.1.1 Wireless Transmission Error Simulation

In NS2.26, WirelessPhy is used to simulate a wireless channel. Mac/802_11 is used to simulate the function of MAC layer. WirelessPhy uses a propagation model to estimate the signal strength of a frame at the receiver. The receiver then uses the following thresholds to decide whether the frame is corrupted.

1. **CSThresh_:** Carrier Sense Threshold. It is used by WirelessPhy of a receiver to determine whether a frame can be detected. If the signal strength of a frame is weaker than CSThresh_, it is discarded by WirelessPhy module and is invisible to Mac/802_11.

2. **RxThresh_:** It is used by WirelessPhy of a receiver to determine whether a frame can be received correctly. If the signal strength of a frame is stronger than RxThresh_, the frame is tagged as correct. Otherwise, the frame is tagged as corrupted and it will be discarded when Mac/802_11 processes this frame.

3. **CPThresh_:** Collision Threshold. When two frames are received simultaneously by Mac/802_11, the signal strength ratio of the stronger frame and the weaker frame is calculated. If it is larger than CPThresh_, the stronger frame will be received correctly and the other frame is ignored. Otherwise, it regards that a collision occurs and the two frames are both discarded.

In order to simulate the error model in section 3.1.2, the receiver needs to calculate SNR of a frame. Signal strength is given by propagation model. We need to figure out the strength of noise and inference. Noise includes noise generated by the receiver and noise from environment. Interference is the signal of frames which are received simultaneously.

Because of the high frequency used by WLAN, environment noise is normally very small. Different environments also have different noise distribution. So, we do not consider environment noise in our project.

For simplicity, instead of calculating noise generated by receiver as described in [54], we deduce noise from receiver sensitivity which is normally provided by manufacturers. Receiver sensitivity is the received signal power with which BER is less than 10^{-5}. To achieve this BER, SNR should be approximately 10dB in WLAN. So we can deduce the noise for different data rates from receiver sensitivity of a card through decreasing receiver sensitivity by 10dB.

As for interference (signal of other frames received simultaneously), it can only be calculated by Mac/802_11 when it tries to detect potential collision. So we calculate SNR at Mac/802_11 by the following equation.

$$SNR = 10 * \log \frac{Rx}{Noise + \sum_{i=1}^{i=NUM-1} Rx_i}$$

Note: NUM is the number of frames received simultaneously. Rx_i is the signal strength of the i^{th} frame at the receiver. It is provided by propagation model. Noise should be calculated from the receiver sensitivity for each data rate.

With SNR calculated by Mac/802_11, we can calculate FER for the frame according to

Equation 3.1. After that, we generate a random number between 0 and 1. If it is less than FER, the frame is tagged as corrupted and discarded at MAC/802_11. By this way, we can simulate wireless transmission error in NS2.

For efficiency, we still keep CSThresh_, RxThresh_, and CPThresh_. CSThresh_ should be set to the noise generated by the receiver with the lowest data rate. RxThresh_ should be less than the receiver sensitivity in order to let frames suffer high BER. CPThresh_ is still 10. By this way, corrupted frames can be discarded early and computation overhead is decreased. Below is the changed procedure for receiving a frame.

1. **WirelessPHY:** If signal strength is less than CSThresh_, the frame is discarded immediately. If signal strength is less than RxThresh_, the frame is tagged as corrupted and sent to MAC. Otherwise, the frame is tagged as correct and sent to MAC.

2. **MAC/802_11:** If other frames arrive simultaneously when a frame is being received, the signal of other frames is added to noise and the SNR is calculated. After that, the FER is calculated according to WLAN error model. Then, a random number is generated. If it is less than FER, the frame is tagged as corrupted. When MAC/802_11 processes this frame, the frame is discarded if the frame is tagged as corrupted. Otherwise, the frame is received correctly. In addtion, MAC/802_11 records the SNR of DATA frame received by it and feeds back this SNR to the sender so that the sender can estimate FER suffered by DATA frame at the receiver.

One module, ErrorModel80211, is added into NS2. It is responsible to calculate SNR and FER for each frame. NS2 users can also configure the noise of each data rate and the

Table 4.1: Propagation Models of NS2

Model	Comments
Free Space	Only the line-of-sight path is considered. It can simulate Satellite links.
TwoRayGround	Both the direct path and a ground reflection path are considered. It can simulate an open space on the ground.
Shadowing	A general model (distance fading, multi-path fading, etc.) It can simulate many environments by adjusting its parameters.

relationship of BER*vs*SNR in ErrorModel80211.

4.1.2 Support for Short Preamble

Since NS2.26 does not support short preamble of IEEE802.11b, we make minor change to support short preamble in NS2.

Firstly, we add a parameter in ErrorModel80211 to enable and disable short preamble. Next, we change the constructor of Mac/802_11. If short preamble is enabled, PHY_MIB of short preamble will be given to Mac/802_11. By this way, Mac/802_11 can correctly calculate transmission time of a frame when short preamble is enabled. Lastly, we also calculate FER according to whether short preamble is enabled.

4.1.3 WLAN Channel in Office

In this thesis, we focus on WLAN in office, the common environment for WLAN channel. In the following paragraphs, we discuss how to set parameters to simulate such a channel.

Table 4.1 shows the propagation models supported by NS2.26 in order to simulate dif-

Table 4.2: Orinoco 80211b PC Card Specification

Transmit Power = 15dBm (0.031622777W)	BPSK (1Mbps)	QPSK (2Mbps)	CCK5.5 (5.5Mbps)	CCK11 (11Mbps)
Receiver Sensitivity(dBm)	-94	-91	-87	-82
Range-Open(m)	550	400	270	160
Range-SemiOpen(m)	115	90	70	50
Range-Closed(m)	50	40	35	25

ferent environments. Thus, Shadowing model should be used in our project.

To be close to the reality, we set other parameters according to a real product, Orinoco 80211b PC Card (Table 4.2 shows its specification). We set pathlossExp_ (path loss exponent) of Shadowing model to 4 so that the signal strength is a little less than the receiver sensitivity at the range of this product. Below is the parameters used in our experiments.

```
ErrorModel80211 noise1    -104
ErrorModel80211 noise2    -101
ErrorModel80211 noise55    -97
ErrorModel80211 noise11    -92
ErrorModel80211 shortpreamble    1
ErrorModel80211 LoadBerSnrFile    ber_snr .txt
Propagation/Shadowing set pathlossExp_    4
Propagation/Shadowing set std_db_    0
Phy/WirelessPhy set L_    1.0
Phy/WirelessPhy set freq_    2.472e9
Phy/WirelessPhy set bandwidth_    11Mb
Phy/WirelessPhy set Pt_    0.031622777
Phy/WirelessPhy set CPThresh_    10
```

Phy/WirelessPhy set CSThresh_ 3.1622777e-14

Phy/WirelessPhy set RXThresh_ 3.1622777e-13

Mac/802_11 set dataRate_ 11Mb

Mac/802_11 set basicRate_ 2Mb

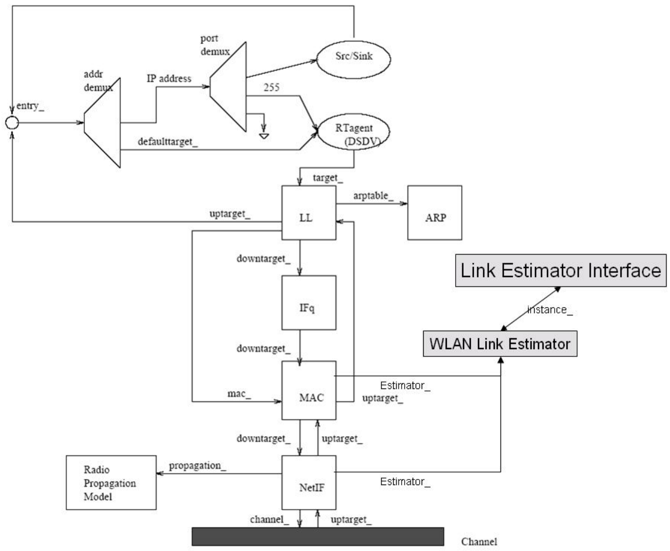

Figure 4.1: Mobile Node Architecture with WLAN Link Estimation Mechanism

4.2 WLAN Link Estimator Implementation in NS2

In order to implement our WLAN link characteristics estimation mechanism, two modules, WLAN Link Estimator and Link Estimator Interface are added into NS2. WLAN Link Estimator is responsible to simulate our WLAN link characteristics estimation mechanism at mobile node and AP. Link Estimator Interface is the unified interface used to provide link characteristics to upper layers. Figure 4.1 shows the architecture of a mobile node that has implemented our link characteristics estimation mechanism.

In NS2, AP is just a mobile node that enables wired routing. WLAN Link Estimator need distinguish AP and normal mobile node. It achieves this goal by comparing node id and base station id. Each mobile node has a unique node-id. We set the node id of AP to base station id and broadcast to all mobile nodes. Thus, WLAN Link Estimator can perform different actions for mobile nodes and AP.

WLAN Link Estimator accepts input from WirelessPhy and Mac/802_11. WirelessPhy reports the period that the channel is idle. WLAN Link Estimator maintains a threshold according to average time wasted by back-off procedure. If the period reported by WirelessPhy is longer than this threshold, WLAN Link Estimator assumes that the channel is really idle during that period.

Mac/802_11 reports collision, frames sent by it, frames received by it, and SNR suffered by data frames sent or received by it. With these information, WLAN Link Estimator implements the function of our mechanism. WLAN Link Estimator of AP deduces the contention status of this channel and broadcasts to all mobile nodes through a Channel Status frame, a control frame added by us. WLAN Link Estimator of a mobile node calculates the link characteristics experienced by it and reports to Link Estimator Interface.

Link Estimator Interface is designed to support adaptation protocols with vertical handoff among multiple wireless interfaces [48]. Different link estimators can be added in stages for different communication medias. Link Estimator Interface provides characteristics of the current link to upper layers. Upper layers need not know the change of network interfaces. Link characteristics can be pulled by consumers or pushed by this interface. It is determined by the consumers. If push method is used, the consumers of link characteristics, such as Reactive TCP, should register to the interface.

Chapter 5

Experimental Results and Discussion

In this chapter, we test the accuracy of our link characteristics estimation mechanism by simulation experiments. Since available bandwidth is one very important characteristics and it is deduced from other link characteristics (LTT, PLR, etc.). So we test the accuracy of our mechanism by comparing the estimated available bandwidth and the measured throughput which is measured by LossMonitor of NS2.

In this chapter, we first describe configurations used in experiments. We then present several typical experiments and their results. Finally, we discuss these experimental results.

5.1 Experiment Setup

In the following experiments, we set the period of AP timer, which triggers when to summarize channel status and broadcast to mobile nodes, to one second. The mobile nodes estimate their link characteristics once per 0.3 second and export these metrics to compare

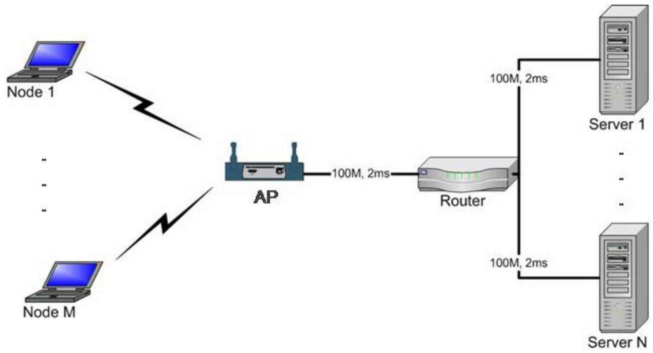

Figure 5.1: Network Topology

with measured values. For each experiment, we simulate for 100 seconds.

In the following experiments, we simulate mobile nodes that use WLAN to access Intranet. Mobile nodes access servers through AP and a router. It is commonly used in wireless office environment. Figure 5.1 shows the network topology used in these experiments. The links between AP, servers and router are all wired links. The bandwidth is 100Mbps and delay is 2ms. The number of mobile nodes and servers may change in different experiments.

5.2 Experiments

In the following paragraphs, we describe several typical experiments and analyze their results. Figures of experiment results are put at the end of this chapter.

5.2.1 One Mobile Sender Moves Around

This experiment is designed to test whether this mechanism can predict ABW when a node moves around and SNR is changing. There is just one mobile node. It sends one 500-

bytes UDP packet to a server every 0.5ms. The sending rate is much higher than theoretical throughput of a WLAN channel simulated by us. By this way, we can measure the maximum throughput. The mobile node moves close to the AP during the simulation. Figure 5.2 shows the measured throughput and estimated ABW during this simulation. Figure 5.3 shows the quality (SNR) of wireless link between this node and AP during this simulation.

The result is very good in this experiment. When the effect of fading is considered (SNR deviation of shadowing model is set to 4), the result is a little different. The estimated ABW is larger than measured throughput, especially when SNR is small. Figure 5.4 shows the measured throughput and estimated ABW, and Figure 5.5 shows the change of SNR over the fading WLAN channel. The non-linear relationship between BER and SNR may be the reason. We use the average SNR of a short period to estimate ABW. Large SNR sample affects the average too much and gives high estimation of wireless link quality. For simplicity, in the following experiments, we focus on a WLAN channel without fading.

5.2.2 Two Saturated Mobile Senders

This experiment is designed to test the effect of contention among two saturated senders. In this experiment, there are two mobile nodes. They send data to an application server through two UDP connections. The length of UDP packet is 500 bytes, and the interval time between packets is 0.5ms. Thus, the two nodes are both saturated nodes. Node 2 is still and begins to send at 1^{st} second. Node 1 begins to send at 15^{th} second. It first moves close to AP, then leaves from AP. Figure 5.6 shows the measured throughput and estimated ABW of the two senders. The result shows that their opportunities of acquiring channel is in inverse proportion to the distance between mobile node and AP. The distance determines its quality of their wireless link with AP.

5.2.3 Two Saturated and Two Unsaturated Mobile Senders

This experiment is designed to test the effect of contention among saturated and unsaturated nodes. In this experiment, there are four mobile senders. They send data to an application server through four UDP connections. Node 1 and node 2 are saturated nodes. Node 3 and node 4 are unsaturated nodes. Node 2 is a stationery host and begins to send 500-bytes packets at 1^{st} second. Node 1 begins to send packets of 500-bytes at 10^{th} second and moves close to AP, then leaves from AP. Node 3 begins to send shorter packets (300-bytes) at 40^{th} second. Node 4 begins to send long packets (800-bytes) at 60^{th} second. Figure 5.7 shows the measured throughput and estimated ABW for the two saturated senders.

5.2.4 One Mobile Receiver Moves Around

This experiment is designed to test our algorithms to estimate link characteristics experienced by a mobile receiver. The mobile node receives data from an application server through one UDP connection. The length of UDP packet is 500-bytes, and the interval is 0.5ms. The sending rate is much higher than theoretical throughput so that we can measure the maximum throughput. The mobile node moves to AP during this experiment. Figure 5.8 shows the measured throughput and estimated ABW of this receiver.

5.2.5 One Sender and Two Receivers

This experiment is designed to test contention among mobile senders and receivers. In this experiment, there are three mobile nodes. Node 3 sends data to an application server through one UDP connection. Node 1 and Node 2 receive data from two application servers through two UDP connections. The data rate of all UDP connections is high enough to make them to be saturated nodes. Node 2 begins to receive packets at 1^{st} second. Node 3 begins to send packets at 20^{th} second. Node 1 begins to receive packets at 40^{th} second. Figure 5.9 shows the measured throughput and estimated ABW of the three nodes. The result shows that the sender contends with AP, and the receivers share bandwidth acquired by AP.

5.3 Discussion

According to above simulation results, the difference of estimated ABW and the measured throughput is normally very small. It means that mobile node which implements our estimation mechanism can accurately estimate the characteristics of a WLAN link. The output of this estimation mechanism can be used by upper layers to improve performance. So, non-intrusive mechanisms to estimate WLAN link characteristics could be used to improve performance of nodes in WLAN.

We notice that the estimated link characteristics lags the changes of contention status on a WLAN channel because AP calculates the contention status periodically based on a timer. The period of the timer should be long enough for AP to correctly deduce contention status of the channel. But it should not bring too much delay to estimated link characteristics. The value should be selected carefully.

We also notice that the estimated link characteristics are not accurate when SNR is small. In consideration of dynamics of wireless link quality, instead of SNR of the latest frame, we estimate link characteristics with the average SNR of frames during a short period. If SNR changes frequently and abruptly, due to the nonlinear relationship of BER*vs.*SNR, wireless link quality represented by the average of SNR is higher than the real link quality. When SNR is small, the change of SNR causes abrupt change of BER, and the problem is more severe. We should measure the curve of BER*vs.*SNR more subtly when SNR is small, and instead of SNR, perhaps BER suffered by the receiver should be fed back to the sender.

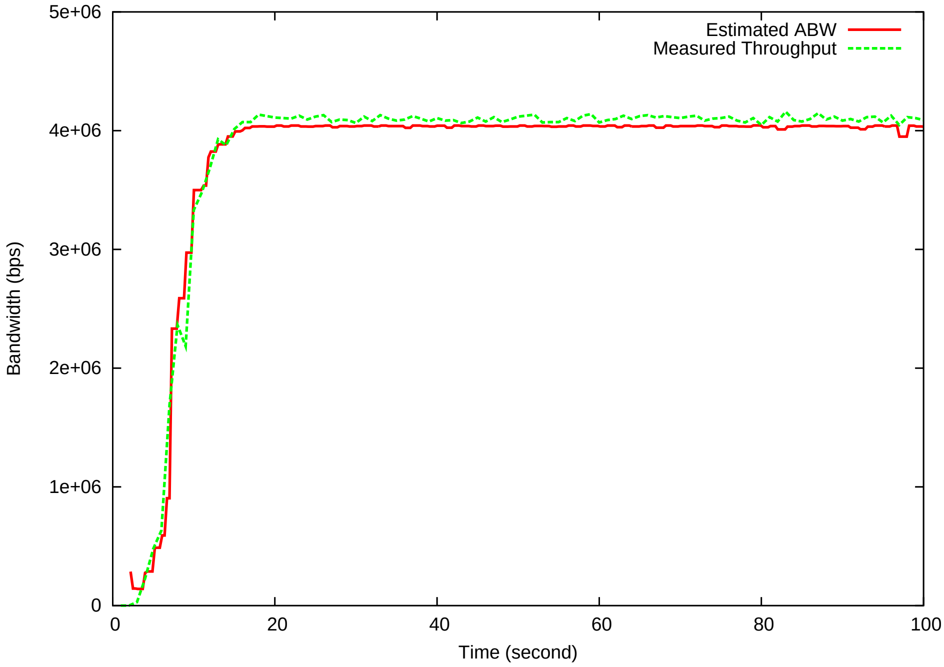

Figure 5.2: One Mobile Sender Moves Around

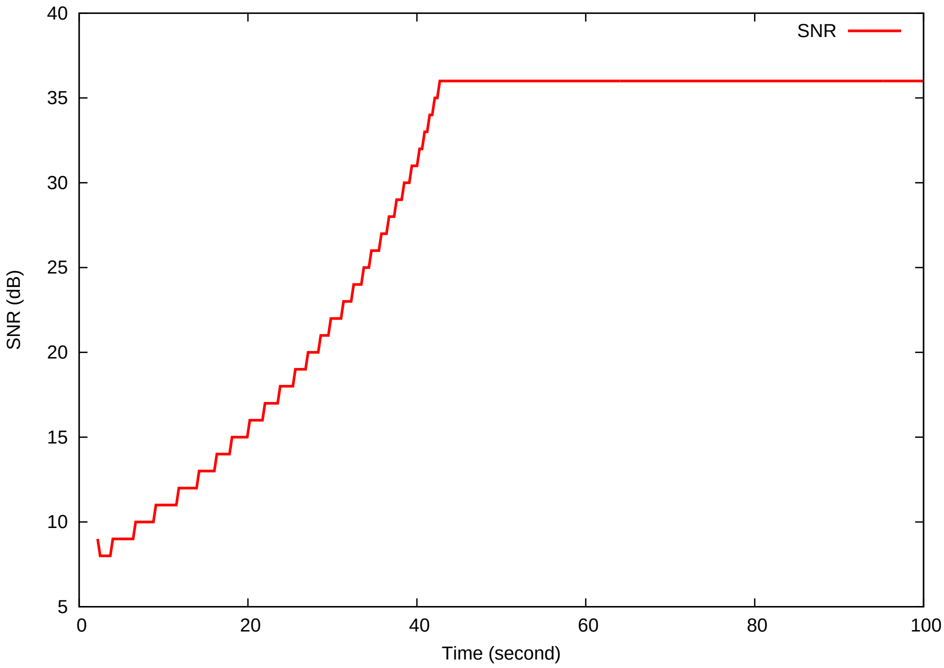

Figure 5.3: SNR of Mobile Sender

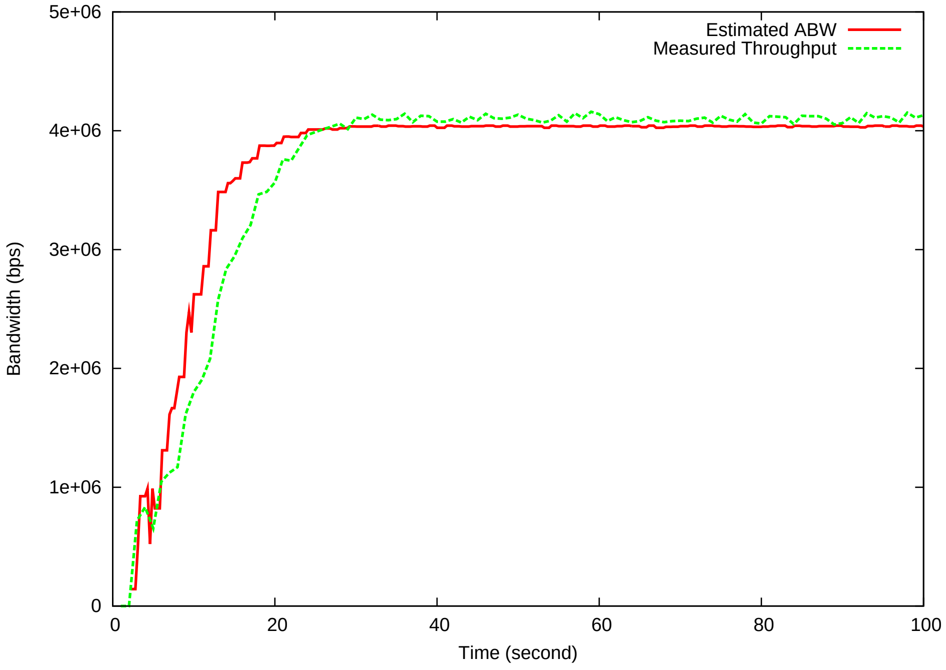

Figure 5.4: One Mobile Sender over Fading Channel

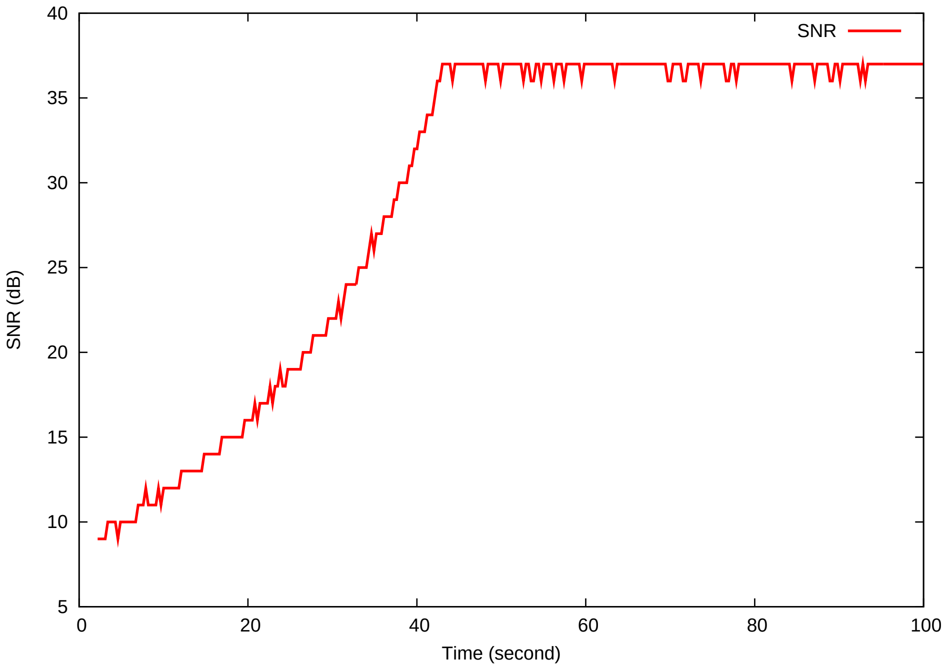

Figure 5.5: SNR of Mobile Sender over Fading Channel

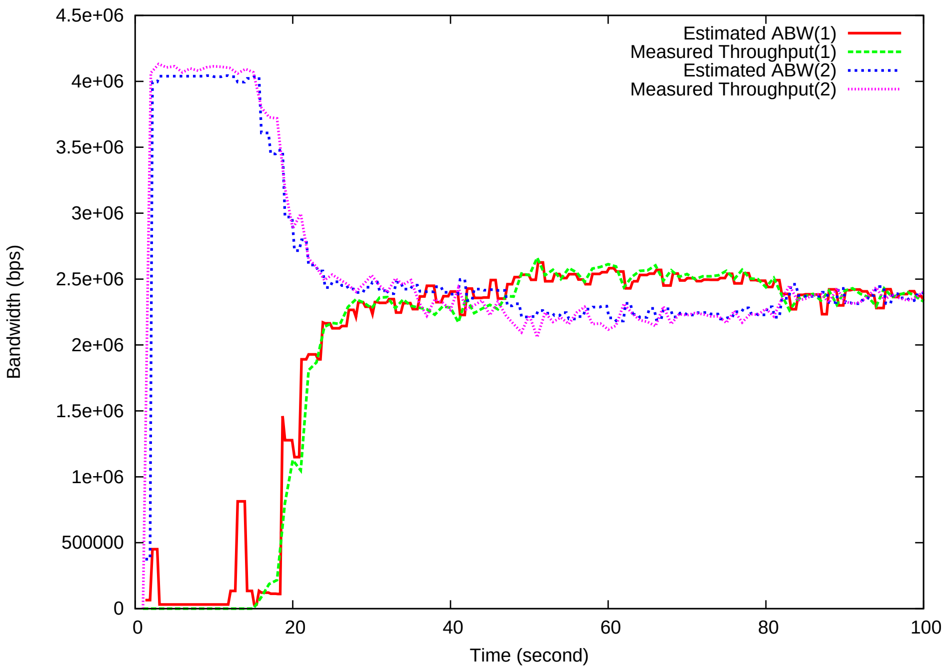

Figure 5.6: Two Saturated Mobile Senders

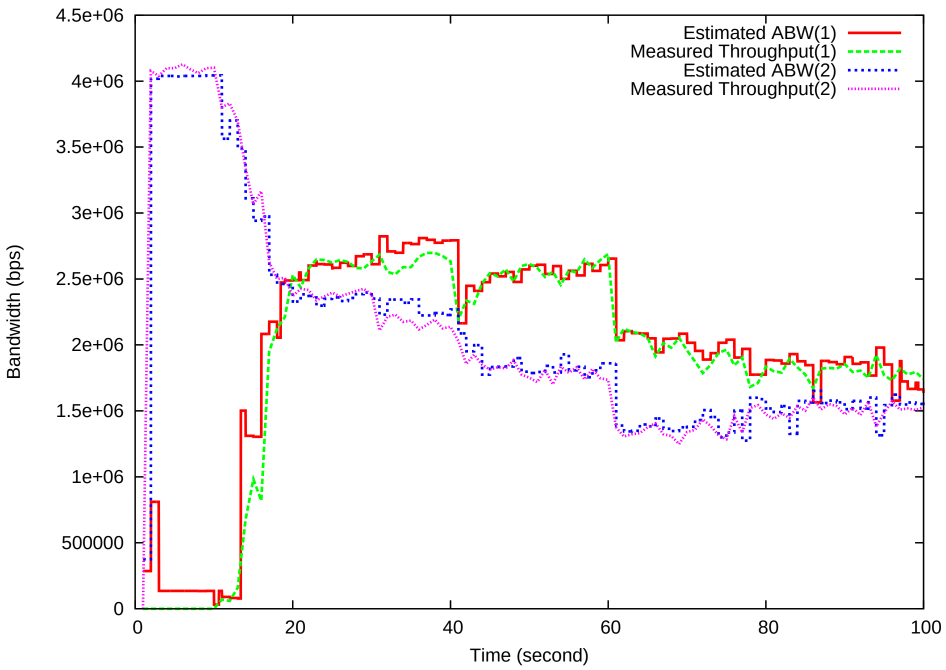

Figure 5.7: Two Saturated Mobile Senders & Two Unsaturated Mobile Senders

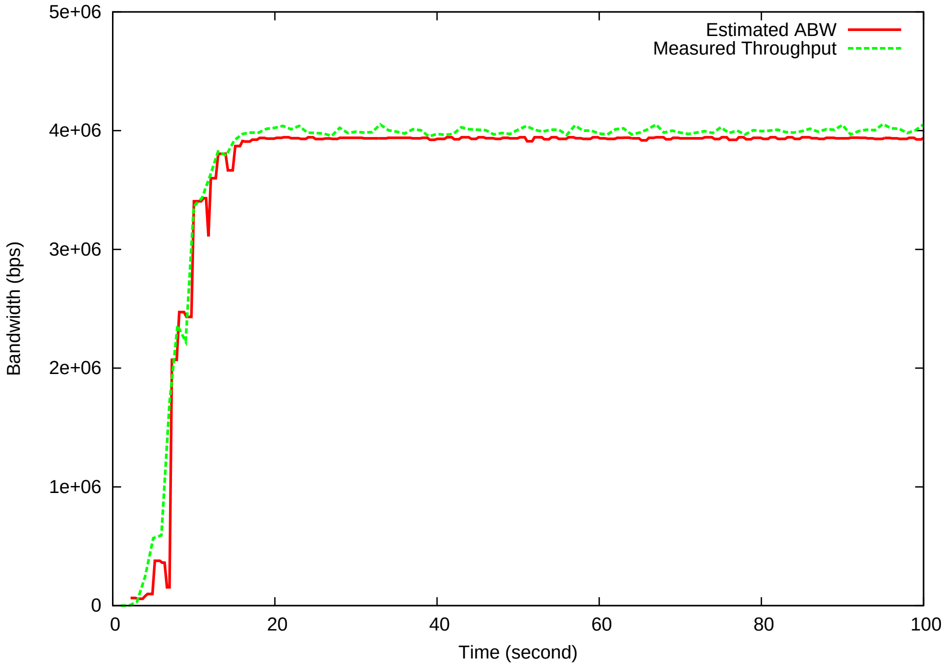

Figure 5.8: One Mobile Receiver

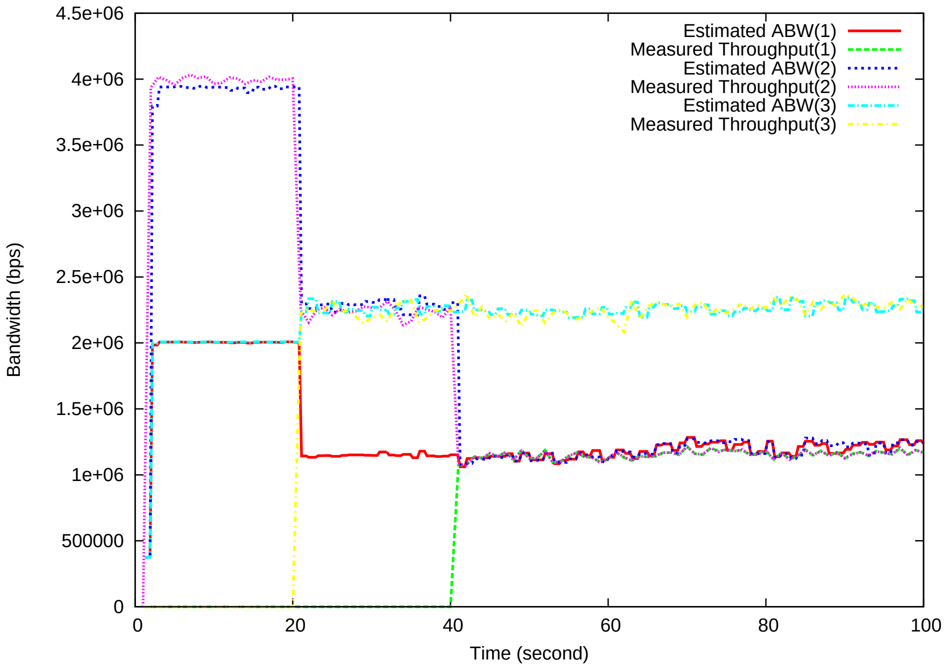

Figure 5.9: One Sender and Two Receivers

Chapter 6

Conclusion

Internet and Intranet play more and more important roles in our life. TCP, perhaps the most widely used transport protocol in Internet and Intranet, was designed for highly reliable links and stationery hosts. It faces many problems when communication links with different characteristics are used.

Especially, wireless links, which are lossy and enable user's mobility, undermine the base of TCP protocol. In addition, many wireless interfaces with different characteristics, may be installed on a mobile node. With the support of mobile IP, a TCP connection may suffer different problems posed by different interfaces. Even only one wireless interface is used, TCP still suffers different link characteristics due to dynamic of wireless link quality and potential contention among mobile nodes.

Since the changing link characteristics bring different problems to TCP at different time, different mechanisms should be used at different time to handle these problems. It is impossible to use a fixed set of TCP mechanisms to achieve optimal performance over wireless links. Reactive TCP, which adopts different mechanisms according to different link characteristics, should be a useful method to improve TCP performance over wireless links. In this thesis, we investigate into how to design Reactive TCP.

Accurate and timely link characteristics estimation algorithms are the prerequisite for the success of Reactive TCP. Due to fading, mobility, and possible contention among mobile nodes, the characteristics of a wireless link may change rapidly and abruptly. Commonly used probing-packets methods are not appropriate because they could not estimate wireless link characteristics accurately and timely with small cost.

In this thesis, we propose a new non-intrusive link characteristics estimation mechanism for IEEE 802.11 DCF based WLAN, one of the most popular wireless access links. We also test its accuracy through simulation experiments. These experimental results show that a mobile node with our estimation mechanism can estimate the characteristics of IEEE 802.11 DCF based WLAN accurately. The output of this estimation mechanism can be used by Reactive TCP to improve performance. So, Reactive TCP with non-intrusive link characteristics estimation mechanism is a feasible solution to improve TCP performance for mobile nodes in wireless networks.

6.1 Summary of Works

In this thesis, we investigate how Reactive TCP should react to network path characteristics. Firstly, we propose an architecture for Reactive TCP. Secondly, we analyze TCP protocol, especially its congestion control, which affects TCP performance very much. This helps how to design an Adaptive TCP Implementation. Thirdly, we summarize network path characteristics, their effects on TCP, and TCP algorithms proposed for different network path characteristics. This gives the rules that should be used by Reactive Engine. Finally, we propose a framework to support Reactive TCP with multiple interfaces.

In this thesis, we propose a new non-intrusive mechanism to estimate link characteristics of IEEE 802.11 DCF based WLAN, one of the most popular wireless access networks. We revise NS2 in order to simulate a channel of IEEE 802.11 DCF based WLAN. We then implement this mechanism in NS2. After that, many experiments are carried out in order to test the accuracy of this mechanism. Based on these fairly good results, we consider that it is possible to estimate wireless link characteristics accurately and timely. This work is expected to be used in the design of Reactive TCP.

6.2 Future Works for Reactive TCP

In this thesis, we investigate into Reactive TCP, one pretty new concept. There are still a lot of work to be done in order to successfully implement a Reactive TCP.

Firstly, instead of designing new algorithms, Reactive TCP adopts different existing algorithms for different network path characteristics in order to enhance TCP performance. Table 2.1 and 2.2 summarizes different algorithms and their proposed network path characteristics. But the effects of algorithms are much more complex. An algorithm proposed for some characteristics may bring adverse effects to other link characteristics, and a link may have all these characteristics. The selection of algorithms should be more careful. For example, large MSS can alleviate ACK congestion on asymmetric link. But it can cause long delay on slow link. Over GPRS, which is slow and asymmetric, large MSS should not be used to avoid ACK congestion. Thus, there are still a lot of work to be done in order to design proper rules for Reactive Engine.

Secondly, a lot of algorithms, such as SACK, need support from both ends. Negotiation is necessary to select algorithms supported by both of them. Currently, TCP just negotiate algorithms one by one, and many algorithms can only be negotiated in SYN segment. It is inefficient and the space of TCP header may be not large enough to negotiate many TCP algorithms. Perhaps a new option is necessary to negotiate the ability of TCP endpoints. In addition, some algorithms, such as Window Scale Option, must be negotiated in SYN segment. This constrain poses some problems in Reactive TCP. For example, when a TCP connection is established through GPRS, Window Scale Option is disabled. But when the user switches to satellite link, Window Scale Option may be necessary to fully utilize the bandwidth provided by satellite link. Thus, these algorithms should be changed in order to be enabled and disabled during a TCP connection.

6.3 Future Works for Wireless Link Estimation

In this thesis, we propose a new mechanism to estimate link characteristics for IEEE802.11 DCF based WLAN. According to simulation results, we consider that it is possible to implement an accurate link characteristics estimation mechanism for IEEE 802.11 DCF based WLAN. There are some future works which are worthwhile to be done.

One work is to implement our link characteristics estimation mechanism on real products with more accurate link quality feed back method. Thus, we can test its accuracy in different wireless environments.

Another work is to design and implement link characteristics estimation mechanisms for other kinds of links. Then we can examine the effectiveness of our architecture when vertical handover [48] occurs. Other WLANs (Bluetooth [10], HiperLAN [4], etc.) can use similar mechanisms to estimate their characteristics. As for General Packet Radio Service (GPRS) [3], an extension to the GSM, the estimation mechanism should be much simpler since a mobile node occupies a wireless channel exclusively.

After Reactive TCP is implemented, we should let Reactive TCP use our link characteristics estimation mechanism and test whether TCP performance can be improved.

Bibliography

[1] 3gpp. Available online at http://www.3gpp.org.

[2] Ns2 network simulator. Available online at http://www.isi.edu/nsnam/ns/.

[3] General packet radio service (gprs) service description, ver. 7.1.0. European Standard 301 344, GSM 03.60, August 1999.

[4] High performance radio local area network, type 2, requirements and architectures for wireless broadband access. TR 101 031 V2.2.1, January 1999.

[5] Wireless lan medium access control (mac) and physical layer (phy) specifications. IEEE P802.11, 1999.

[6] Wireless lan medium access control (mac) and physical layer (phy) specifications—high-speed physical layer in the 5 ghz band. IEEE P802.11, 1999.

[7] Wireless lan medium access control (mac) and physical layer (phy) specifications—higher-speed physical layer extension in the 2.4ghz band. IEEE P802.11, 1999.

[8] Gsm ts, radio link protocol(rlp) for data and telematic services on the mobile station, January 2000.

[9] Rlc protocol specification. 3G TS 25.322, V3.2.0, March 2000.

[10] Specification of the bluetooth system version 1.1, 2001.

[11] Calveras A., Linares J., and J. Paradells. Window prediction mechanism for improving tcp in wireless asymmetric links. In *Proc. IEEE Global Communications Conference (GLOBECOM)*, 1998.

[12] M. Allman, H. Balakrishnan, and S. Floyd. Enhancing tcp's loss recovery using limited transmit. RFC 3042, January 2001.

[13] M. Allman, S. Floyd, and C. Partridge. Increasing tcp's initial window. RFC 3390, 2002.

[14] M. Allman, D. Glover, and L. Sanchez. Enhancing tcp over satellite channels using standard mechanisms. RFC 2488, 1999.

[15] M. Allman, V. Paxson, and W. Stevens. Tcp congestion control. RFC 2581, April 1999.

[16] Mark Allman. On the generation and use of tcp acknowledgments. *Computer Communication Review*, 28, October 1998.

[17] Mark Allman. On the generation and use of tcp acknowledgments. *ACM Computer Communication Review*, 28, 1998.

[18] A. Bakre. I-tcp: indirect tcp for mobile hosts. In *in Proceedings of the 15th International Conference on Distributed Computing Systems (ICDCS'95)*, 1995.

[19] H. Balakrishnan, R. H. Katz, and V. N. Padmanabhan. The effects of asymmetry on tcp performance. *Mobile Networks and Applications*, 4, October 1999.

[20] H. Balakrishnan, V. N. Padmanabhan, G. Fairhurst, and M. Sooriyababdara. Tcp performance implications of network path asymetry. RFC 3449, 2002.

[21] H. Balakrishnan, V. N. Padmanabhan, G. Fairhurst, and M. Sooriyabandara. Tcp performance implications of network path asymmetry. RFC 3449, 2002.

[22] H. Balakrishnan, S. Seshan, E. Amir, and R.H. Katz. Improving tcp/ip performance over wireless networks. In *Proceedings of Mobicom95*, 1995.

[23] Vaduvur Bharghavan, Alan J. Demers, Scott Shenker, and Lixia Zhang. MACAW: A media access protocol for wireless LAN's. In *SIGCOMM*, pages 212–225, 1994.

[24] Ethan Blanton and Mark Allman. On making tcp more robust to packet reordering. *ACM SIGCOMM Computer Communication Review*, 32, January 2002.

[25] J. Border, M. Kojo, J. Griner, G. Montenegro, and Z. Shelby. Performance enhancing proxies intended to mitigate link-related degradations. RFC 3135, June 2001.

[26] R. Braden. T/tcp – tcp extensions for transactions functional specification. RFC 1644, July 1994.

[27] Lawrence S. Brakmo, Sean W. O'Malley, and Larry L. Peterson. Tcp vegas: New techniques for congestion detection and avoidance. In *SIGCOMM*, 1994.

[28] Lawrence S. Brakmo and Larry L. Peterson. Tcp vegas: End to end congestion avoidance on a global internet. *IEEE Journal on Selected Areas in Communications*, 13:1465–1480, 1995.

[29] Claudio Casetti, Mario Gerla, Saverio Mascolo, M.Yahya Sanadidi, and Ren Wang. Tcpwestwood: End-to-end bandwidth estimation for enhanced transport over wireless links. *Wireless Networks*, 8:467–479, 2002.

[30] P. Chou and Z. Miao. Rate-distortion optimized streaming of packetized media. Technical report, February 2001.

[31] S. Dawkins, G. Montenegro, M. Kojo, and V. Magret. End-to-end performance implications of slow links. RFC 3150, 2001.

[32] S. Dawkins, G. Montenegro, M. Kojo, V. Magret, and N. Vaidya. End-to-end performance implications of links with errors. RFC 3155, 2001.

[33] S. Floyd and T. Henderson. The newreno modification to tcp's fast recovery algorithm. RFC 2582, April 1999.

[34] Sally Floyd, Jamshid Mahdavi, Matt Mathis, and Matt Podolsky. An extension to the selective acknowledgment (sack) option for tcp. RFC 2883, July 2000.

[35] Andrei Gurtov and Reiner Ludwig. Responding to spurious timeouts in tcp. In *IEEE INFOCOM 2003*, 2003.

[36] H. Inamura, G. Montenegro, R. Ludwig, A. Gurtov, and F. Khafizov. Tcp over second (2.5g) and third (3g) generation wireless networks. RFC 3481, 2003.

[37] Intersil. *HFA3861B; Direct Sequence Spread Spectrum Baseband Processor*, January 2000.

[38] Ming-Chit I.T., Jinsong D., and W. Wang. Improving tcp performance over asymmetric networks. *ACM Computer Communications Review (CCR)*, 30, 2000.

[39] Postel J. Internet protocol. RFC 791, September 1981.

[40] Postel J. Transmission control protocol - darpa internet program protocol specification. RFC 793, September 1981.

[41] V. Jacobson, R. Braden, and D. Borman. Tcp extensions for high performance. RFC 1323, May 1992.

[42] Van Jacobson. Congestion avoidance and control. In *ACM SIGCOMM*, 1988.

[43] Van Jacobson. pathchar—tool to infer characteristics of internet paths. Presented at the Mathematical Sciences Research Institute, April 1997. Available online at ftp://ftp.ee.lbl.gov/pathchar/.

[44] M. Jain and C. Dovrolis. End-to-end available bandwidth: measurement methodology, dynamics, and relation with tcp throughput. In *Proc. ACM SIGCOMM*, 2002.

[45] Y. Y. Kim and San qi Li. Modeling multipath fading channel dynamics for packet data performance analysis. *Wireless Networks*, 6:481–492, 2000.

[46] Reiner Ludwig and Randy H. Katz. The eifel algorithm: making tcp robust against spurious retransmissions. *ACM SIGCOMM Computer Communication Review*, 30, January 2000.

[47] Stemm Mark. Vertical handoffs in wireless overlay networks. Technical report, 1996.

[48] Stemm Mark. Vertical handoffs in wireless overlay networks. Technical report, 1996.

[49] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow. Tcp selective acknowledgment options. RFC 2018, 1996.

[50] G. Montenegro, S. Dawkins, M. Kojo, V. Magret, and N. Vaidya. Long thin networks. RFC 2757, 2000.

[51] M. Mouly and M. B. Pautet. The gsm system for mobile communications. Europe Media Duplication, 1993.

[52] John Nagle. Congestion control in ip/tcp internetworks. RFC 896, January 1984.

[53] Vern Paxson. End-to-end internet packet dynamics. In *ACM SIGCOMM*, 1997.

[54] Theodore S. Rappaport. *Wireless Communications: PRINCIPLES AND PRACTICE*. Prentice Hall, second edition, 2002.

[55] V. Ribeiro, M. Coates, R. Riedi, S. Sarvotham, B. Hendricks, and R. Baraniuk. Multifractal cross-traffic estimation. In *Proc. of ITC Specialist Seminar on IP Traffic Measurement*, September 2000.

[56] Vinay J. Ribeiro, Rudolf H. Riedi, Jiri Navratil, Les Cottrell, and Richard G. Baraniuk. pathchirp: Efficient available bandwidth estimation for network paths. In *Proceedings Workshop on Passive and Active Measurement*, 2003.

[57] Balan R.K, B.P.Lee, K.R.R.Kumar, L.Jacob, W.K.G.Seah, and A.L.Ananda. Tcp hack: a mechanism to improve performance over lossy links. *Computer Networks*, 39:347–361, 2002.

[58] B. Sadegi, V. Kanodia, A. Sabharwal, and E. Knightly. Oppurtunistic media access for multirate ad hoc networks. In *International Conference on Mobile Computing and Networking(MOBICOM'02)*, 2002.

[59] Jeffrey Semke, Matthew Mathis, and Jamshid Mahdavi. Automatic tcp buffer tuning. In *in ACM SIGCOMM 98 Proceedings 1998*, 1998.

[60] Srikant Sharma. Analysis of 802.11b mac: A qos, fairness, and performance perspective, 2003.

[61] B. Sklar. *Digital Communication Fundamentals and Applications*. Englewood Cliffs, NJ., 1998.

[62] Pradeep Sudame and B.R. Badrinath. On providing support for protocol adaptation in mobile wireless networks. *Mobile Networks and Applications*, 6:43–55, January 2001.

[63] Mineo Takai, Jay Martin, and Rajive Bagrodia. Effects of wireless physical layer modeling in mobile ad hoc networks, 2004.

[64] Y.C. Tay and K.C. Chua. A capacity analysis for the ieee 802.11 mac protocol. *Wireless Networks*, 7:159–171, 2001.

[65] Jian Zhang, Liang Cheng, and Ivan Marsic. Models for non-intrusive estimation of wireless link bandwidth. In *Personal Wireless Communication Conference*, Venice, Italy, September 2003.