Error Recovery Schemes in Wireless Data Networks

by

Oguz Silahtar

Submitted to the Department of Electrical Engineering and Computer Science
in Partial Fulfillment of the Requirements for the Degree of
Master of Engineering in Electrical Engineering and Computer Science
at the Massachusetts Institute of Technology jointly with QUALCOMM Incorporated

May 2001

Author_____

Department of Electrical Engineering and Computer Science
May 21, 2001

Certified by____

Hari Balakrishnan
MIT Thesis Supervisor

Certified by_____

Bibhu Mohanty
QUALCOMM Thesis Supervisor

Accepted by_____

Arthur C. Smith
Chairman, Department Committee on Graduate Theses

Error Recovery Schemes in Wireless Data Networks

by

Oguz Silahtar

oguz@alum.mit.edu

Submitted to the
Department of Electrical Engineering and Computer Science

May 21, 2001

In Partial Fulfillment of the Requirements for the Degree of
Master of Engineering in Electrical Engineering and Computer Science
at the Massachusetts Institute of Technology jointly with QUALCOMM Incorporated

Abstract

Wireless data communication networks have many protocol layers and implement various methods of error recovery schemes. In order to maximize performance, it is important to minimize possible inefficient cross layer interactions. In this thesis the forward link of a CDMA2000 High Data Rate (HDR) system was analyzed using OPNET and physical layer simulation tools. The trade-off between error recovery at the physical layer and the radio link protocol (RLP) was studied for various channel conditions and a target packet error rate for optimal system performance was found. In addition, the HDR system layers were characterized. The issues of interest were the throughput, packet error rate, and delay. A simplified model with few well-defined interfaces between these layers was used to explain system behavior.

Thesis Supervisor: Hari Balakrishnan
Title: Assistant Professor

Acknowledgments

This thesis is dedicated to my parents. They have supported me in every way.

First, I would like to thank both my supervisors. At Qualcomm, Bibhu Mohanty provided guidance and mentorship all along the way. It was a pleasure working with him, and I am grateful for all his help. Also, I would like to thank my supervisor at MIT, Prof. Hari Balakrishnan, for his encouragement and advice.

I am also grateful for Qualcomm for providing the resources necessary to complete this thesis, as well as my previous internship assignments. I thank the entire Systems Group in Corporate Research and Development for all their support; and the Hardware Group for creating a friendly environment of which Stefanie Roesslein deserves special mention.

I want to express my gratitude to Peter Agboh, who also worked on his thesis at Qualcomm at the same time. He was helpful as a friend as well as colleague. I would like to thank the staff of the Computer System Engineering class at MIT, for providing me wonderful TA experience. Finally I would like to thank all my friends at MIT for making my stay here enjoyable.

Table of Contents

ACKNOWLEDGMENTS 3

LIST OF FIGURES 5

LIST OF TABLES 6

CHAPTER 1 : INTRODUCTION 7

1.1 MOTIVATION 7
1.2 CONTRIBUTIONS 8
1.3 ROADMAP OF THE THESIS 10

CHAPTER 2 : BACKGROUND 11

2.1 TCP PROTOCOL 11
2.2 CHALLENGES IN THE HDR WIRELESS SYSTEM 15
2.3 THE HDR SYSTEM OVERVIEW 18
2.3.1 System Overview *18*
2.3.2 The Architecture *19*
2.3.3 The Physical Layer *21*
2.3.4 The Scheduler *25*
2.3.5 The DRC mechanism *26*
2.3.6 The Physical Layer ARQ Mechanism *27*
2.3.7 The Radio Link Protocol *30*
2.3.8 Overview *34*
2.3.9 Delays in the System *35*
2.3.10 Overheads in the System *41*
2.3.11 Throughput Ratios for different layers of the system *42*

CHAPTER 3 : THE SIMULATION ENVIRONMENT 46

3.1 OPNET SIMULATION 47
3.2 PHYSICAL LAYER SIMULATIONS 53
3.3 REAL LIFE DATA 55

CHAPTER 4 : DISCUSSION OF RESULTS 56

4.1 THROUGHPUT VERSUS PACKET ERROR RATE AND RLP PERSISTENCY 56
4.2 FRAMEWORK AND EMPIRICAL MODEL 61
4.2.1 Effect of Target Packet Error Rate on Physical Layer Characteristics *63*
4.2.2 TCP throughput with PPER and RLP Persistency *70*

CHAPTER 5 : CONCLUSION AND FUTURE WORK 74

BIBLIOGRAPHY 76

APPENDIX 79

7.1 CHANNEL MODELS 79
7.2 DRC DISTRIBUTIONS FOR DIFFERENT CHANNEL CONDITIONS 81

List of Figures

FIGURE 1.2-1: MODEL OF THE SYSTEM 9
FIGURE 2.3-1: DIFFERENT LAYERS IN T HE HDR-INTERNET 19
FIGURE 2.3-2: ARCHITECTURE REFERENCE MODEL 19
FIGURE 2.3-3: FORWARD CHANNEL STRUCTURE 25
FIGURE 2.3-4: FORWARD LINK SLOT STRUCTURE 25
FIGURE 2.3-5: DRC TIMING FOR GATED TRANSMISSION 29
FIGURE 2.3-6: MULTI-SLOT PHYSICAL LAYER PACKET WITH NORMAL TERMINATION 29
FIGURE 2.3-7: MULTI-SLOT PHYSICAL LAYER PACKET WITH EARLY TERMINATION 30
FIGURE 2.3-8: DEFAULT PACKET APPLICATION ENCAPSULATION 33
FIGURE 2.3-9: RLP TRANSMIT SEQUENCE NUMBER VARIABLE 33
FIGURE 2.3-10: RLP RECEIVE SEQUENCE NUMBER VARIABLES 33
FIGURE 2.3-11: HDR SYSTEM DELAY REFERENCE MODEL 36
FIGURE 2.3-12: OVERVIEW OF THE LAYER THROUGHPUTS ANALYZED 42
FIGURE 2.3-14: RATIO OF TCP THROUGHPUT TO RLP THROUGHPUT 43
FIGURE 2.3-16: RATIO OF RLP LAYER THROUGHPUT TO PHYSICAL LAYER THROUGHPUT 44
FIGURE 2.3-18: PHYSICAL LAYER THROUGHPUT USED FOR RLP/PHYSICAL LAYER THROUGHPUT 44
FIGURE 2.3-20: RATIO OF TCP THROUGHPUT TO PHYSICAL LAYER THROUGHPUT 45
FIGURE 2.3-1: SIMULATION OVERVIEW 47
FIGURE 3.1-1: OPNET NETWORK MODEL 48
FIGURE 3.1-2: MPC NODE MODEL 50
FIGURE 3.1-3: MPT NODE MODEL 50
FIGURE 3.1-4: AT64 NODE MODEL 50
FIGURE 3.1-5: THE GENERAL MODEL USED IN OPNET 51
FIGURE 3.1-6: THE MPC MODEL USED IN OPNET 51
FIGURE 3.1-7: THE MPT MODEL USED IN OPNET 51
FIGURE 3.2-1: PER PERFORMANCE FOR HDR CODES 54
FIGURE 4.1-1: TCP THROUGHPUT AS A FUNCTION OF TPER AND RLP 57
FIGURE 4.1-2: SIMULATED THROUGHPUT GAIN IN KM30 CHANNEL 58
FIGURE 4.1-3: THE THROUGHPUT GAIN, BY USING EARLY TERMINATION 59
FIGURE 4.1-4: THROUGHPUT GAIN WITH TPER FOR KM30 WITH EARLY TERMINATION 60
FIGURE 4.2-1: BREAKING UP THE TCP THROUGHPUT 62
FIGURE 4.2-2: THE MECHANISMS THAT INFLUENCE PHYSICAL LAYER CHARACTERISTICS. 63
FIGURE 4.2-3: THE INCREASE OF THE DRC VALUES AS THE TPER INCREASES. 65
FIGURE 4.2-4: THE PPER AS THE TPER CHANGES. 65
FIGURE 4.2-5: PER-DATA RATE PLOT , FROM A PER-SINR PLOT 66
FIGURE 4.2-6: PERCENTAGE OF SLOTS FOR WHICH THE DRC INCREASED 68
FIGURE 4.2-7: PERCENTAGE OF SLOTS FOR WHICH THE DRC DECREASED 68
FIGURE 4.2-8: PERCENTAGE OF SLOTS FOR WHICH THE DRC STAYED THE SAME 68
FIGURE 4.2-9: PER FOR THE SLOTS FOR WHICH THE DRC INCREASED 69
FIGURE 4.2-10: PER FOR THE SLOTS FOR WHICH THE DRC STAYED THE SAME 69
FIGURE 4.2-11: PER FOR THE SLOTS FOR WHICH THE DRC DECREASED 69
FIGURE 4.2-12: THROUGHPUT VERSUS PHYSICAL LAYER PACKET ERROR RATE AND RLP PERSISTENCY. 71
FIGURE 4.2-13: TCP THROUGHPUT FOR THE KM30 CHANNEL 72
FIGURE 4.2-14: ESTIMATED THROUGHPUT GAIN IN KM30 CHANNEL 73
FIGURE 7.1-1: EMBEDDED SECTOR SETUP USED IN CHANNEL SIMULATIONS 80

List of Tables

TABLE 1: MODULATION PARAMETERS FOR THE FORWARD TRAFFIC CHANNEL AND THE CONTROL CHANNEL (PART 1 OF 2) 22
TABLE 2: MODULATION PARAMETERS FOR THE FORWARD TRAFFIC CHANNEL AND THE CONTROL CHANNEL (PART 2 OF 2) 23
TABLE 3: PREAMBLE REPETITION 24
TABLE 4: COMPONENTS OF THE FORWARD LINK PHYSICAL LAYER DELAY 37
TABLE 5: COMPONENTS OF THE MPC-MPT TRANSPORT DELAY 37
TABLE 6: COMPONENTS OF HDR SYSTEM FORWARD LINK DELAY 38
TABLE 7: COMPONENTS OF THE REVERSE LINK PHYSICAL LAYER DELAY 39
TABLE 8: COMPONENTS OF HDR SYSTEM REVERSE LINK DELAY IN THE RELAY MODEL CONFIGURATION 40
TABLE 9: OVERHEADS IN THE SYSTEM 41
TABLE 10: CHANNEL SIMULATION PARAMETERS 79

Chapter 1: Introduction

1.1 Motivation

In the last decade cellular phone use has become widespread. The early uses of these devices were mainly the same as their landline counterparts. As the importance of data became clear and mobile computing devices grew more common, data communications emerged as the next application for this platform. However, data and voice have inherently different requirements and characteristics. In a voice system, the human ear can cope with a moderate amount of error, but any delay greater than 100 ms introduces discomfort. On the other hand data applications require fairly low error rates and are more tolerant of higher latencies. All the wireless systems deployed up date were designed and optimized for voice applications; thus the data services that service providers offer are basically enhancements to the existing wireless networks. When a data application is run over these existing systems many inefficiencies surface. This was the motivation for developing Cdma20001xEV-DO (HDR).

There are several error recovery schemes in a wireless data network such as HDR. At the lowest layer the coding used in the physical layer is a type of error recovery scheme. All the physical-layer packet transmissions use the same chip rate for modulating the base-band signal. However, the different data rates are just different ways of coding, repeating and framing these base-band bits. Naturally, a low data rate makes use of codes with stronger error recovery, and uses more repetition. If a packet is transmitted successfully with fewer repetitions than planned, the physical layer ARQ scheme terminates the transmission, avoiding redundant repetitions. This can be thought of as an enhancement to the error recovery scheme at the physical layer. The second error recovery scheme is at the Radio Link Protocol (RLP)

layer. The RLP layer provides a limited number of retransmissions for missing packets requested by negative acknowledgements. The final end-to-end error recovery scheme is part of TCP. The error rates that each of these protocols provide to the upper layers can be adjusted. At the physical layer this can be achieved by choosing a different coding scheme for the physical layer packets. At the RLP layer the error rate can be adjusted by choosing a different negative acknowledgement policy at the receiver and retransmission policy at the sender. TCP is practically a reliable protocol, and unless the network connection is lost for an extended period, the packets are eventually transmitted successfully. However TCP controls the packet-flow and influences the throughput of the system.

There are two main motivations of this thesis. The practical motivation is to improve the TCP layer throughput of the wireless data system. The second motivation is to characterize the system and come up with a simplified model, with few parameters and few well-defined interfaces between the different layers.

1.2 Contributions

At the physical layer of HDR a packet error rate of 0.02 was targeted, mainly due to the intuition of the people designing the system and partly due to the existence of the codes used in voice systems. This error rate is too high for TCP to function efficiently, so at the RLP layer single retransmissions based on single negative acknowledgements were implemented, lowering the packet error rate TCP sees to 0.001. For good performance of TCP, this is probably the desired error rate. However, the same error rate can be achieved by a different combination of error recovery schemes at the physical and RLP layer. Our main contribution is the study of the trade-off between error control at these two layers. We showed that having a

more persistent RLP Protocol above a more aggressive, high error rate - high throughput physical layer does not noticeably improve the throughput of the system.

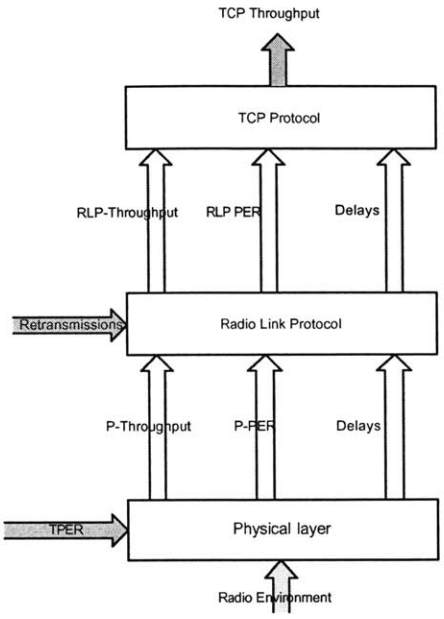


Figure 1.2-1: Model of the system

The second important contribution of our work is in the analysis of the system and characterization of the different layers. We divided the system into three layers (Physical, RLP, TCP) as in Figure 1.2-1 for analysis and determined the important parameters of each layer. We characterized the physical layer for different channel conditions in terms of packet error rate and throughput. We also analyzed the sensitivity of the TCP throughput to physical error rate through simulations, and found that it decreases starting from a packet error rate of 0.05 with a single retransmission RLP scheme, and at higher error rates with more persistent RLP designs. The characterizations of the different layers were used as empirical models, and combined to show that the interactions defined between the layers are reasonably complete. This characterization and model of the system is the second important contribution of this research.

1.3 Roadmap of the Thesis

The rest of the thesis is organized as followed. Chapter 2 presents the necessary background in TCP and HDR. Chapter 3 explains the simulation environment used in the work. Chapter 4 presents the results and discusses their meanings. Finally Chapter 5 summarizes our conclusions and contributions, and discusses some future work. There is also an appendix with some technical simulation parameters and detailed analysis for the very interested readers.

Chapter 2: Background

In this chapter we will go over the background material necessary for this work. In general we provide the intuition behind the protocols and mechanisms. Section 1 begins with an overview of TCP and section 2 introduce possible inefficient interactions between TCP and the wireless system. Section 3 is a brief overview of the HDR system.

2.1 TCP Protocol

There are many applications based on data and many devices that these applications can run on. Making all these compatible with each other is not an easy task. The layered architecture is designed to solve this problem. In this design the system is divided into layers that do certain functions, and have clear interfaces for other layers Figure 2.3-1). The Internet today uses the Internet Protocol Suite that defines the layers above data link layer. The network protocol of this suite is *IP, the Internet Protocol.* There are many possible transport protocols, the most common of which is TCP[1]. The IP protocol provides fragmentation and routing for data packets. TCP provides congestion control and reliable data transmission. Both of these protocols were designed before wired data communication networks, and they were designed to work well in low error rate environments. Since the wireless telecommunication networks designed for voice do not provide this kind of an environment, the IP suite runs into problems and inefficiencies. TCP controls the flow of the data transmission, to be able to adapt to changing network conditions, such as delay, data rate and congestion. It achieves this by controlling the amount of data in flight in the connection between the sender and the receiver.

It segments the data and assigns sequence numbers before giving them off to the IP layer. TCP also manages the reliability of the transmissions by the use of a *cumulative ACK* scheme. The receiver sends *acknowledgements* to tell the sender that it received packets correctly. If the receiver sends an ACK for sequence number *n*, this means that it has received all bytes (octets) with sequence number less than *n*. This way the receiver does not have to send an ACK for every packet and TCP will function well even if an ACK is lost once in a while. With the aid of these ACKs, TCP detects transmission failures and retransmits the necessary packets to provide reliability.

The congestion control is achieved by a window-based scheme. At any time there is a window of sequence numbers that the sender is allowed to transmit. Ideally the sender would make the most of the available connection, by sending just enough data to keep the connection between itself and the receiver fully occupied. If the connection data rate[2] is *r* and the one-way delay is *d*, the ideal sender would send packets at a rate *r*. TCP achieves this congestion control by using a *self-clocking* mechanism. The sender sends a new packet for every ACK it receives, thus it conserves the number of packets in flight. In our ideal scenario by the time the ACK for a packet is received by the sender, it has sent *2*d*r (bandwidth delay product)* packets.

Unfortunately the world is not this simple, and the data rate of the connection keeps changing with time. Thus the sender needs to constantly adjust the rate at which it is sending data. TCP adjusts the sender's data rate by limiting the allowed packets to be transmitted, to a window of sequence numbers. If the last ACK verified the successful transmission of packet *n*, the sender can send the next *w* packets after *n*. The value of *w* is the minimum of two values: the *receiver window size*, and *congestion window size*. The receiver window size is announced

---

[1] Here I will try to build intuition about the workings of TCP. For a full coverage of the topic the reader can refer to one of the sources listed in the bibliography.
[2] By connection data rate I mean the bottleneck data rate; if the connection consists of more than one data link the lowest of all the rates of the links.

by the receiver and ensures that the sender is not sending data at a faster rate than the receiver can process. The sender adapts to different network conditions by changing the congestion window size *cwnd*. In the beginning of a connection the goal is to find the right window size as quickly as possible. The window starts with a very low size of one and increases by 1 for every ACK received; this phase is called *slow start*[3], and causes cwnd to double every round-trip time[4][19]. Once the window size reaches a threshold *ssthresh* the slow start mechanism stops and *congestion avoidance* phase starts. In the congestion avoidance phase the window increases linearly with time; for every ACK received, the window size is increased by *1/(window size)*, thus for every round trip interval the window size is increased by *1*.

Since there is a mechanism to increase the window size there needs to be a mechanism to decrease it, when the network conditions deteriorate. There can be two unwanted events in a TCP connection, and neither of them can be prevented. The first case is a *packet loss* not due to congestion. In this case some packet is lost since no communication link is fully reliable and the sender should just retransmit the lost packet. The second event is a congestion of the network, this is a much more severe condition, since it results in many packets getting dropped. It also means that the sender is sending packets at a faster rate than the connection can support,[5] so unless the sender changes its behavior, more packets will be dropped.

TCP sender should react to a congestion situation by retransmitting all the packets in its window, since a good amount of them were lost, and reducing its window size. As we can see these two events are essentially different and require different remedies. In the first case the sender simply retransmits the lost packet(s). In case of a congested network, it reduces its *cwnd* to its minimum value forcing itself to retransmit all the packets in flight; and halves the

[3] Starting off with a large window could easily cause congestion in low bandwidth or highly shared links.

[4] Not if there are delayed acknowledgements

[5] To be more precise; a link in the connection is being flooded by too many packets, by all the connections that utilize that link.

value of *ssthresh,* reducing its estimate of available bandwidth in the network. The way TCP detects these events is by monitoring the ACK mechanism mentioned earlier.

The flow of incoming ACKs can be interrupted in two ways: they might stop coming, or *duplicate ACK*s might be received.[6] If the sender does not receive any ACKs at all for some time *RTO* this is a clear sign of congestion in the network, so the sender should react accordingly. Here the problem is that the network could have slowed down for a short time and for some reason the packets could have been delayed more than usual, so it is important to choose the value of the *retransmission timeout (RTO) appropriately.* It is evident that too small of a *RTO* value would cause the sender to wait unnecessarily, and too small of a value would trigger the congestion control mechanism when there is no congestion. Luckily TCP is not helpless is this decision; it monitors the *round trip time (rtt)* of packets by checking when the packets were sent and the corresponding ACKs were received. The *rto value* should be a combination of the mean round trip time, and the variation of the round trip time.[7] The values are calculated by using a causal estimator with an exponential impulse response.[8] There exact formulas for these values vary between different TCP implementations, but most of them work reasonably well at detecting network congestion.

When detecting packet losses that are not due to congestion, TCP does not make use of a timer; instead the sender observes the cumulative ACKs sent by the receiver. Since the ACKs are cumulative, if a packet is lost the receiver sends the same ACK for every packet it receives until it receives the missing packet. This suggests that duplicate ACKs could be monitored to detect packet losses. The only complication is due to packet reordering in the network. In the

---

[6] These correspond to the two possible ways a sender can become aware of a lost packet or congestion: The receiver can explicitly send a message, or the sender can keep track of timer. The timer method is guaranteed to catch all link problems, but the first method is almost always faster. A protocol like TCP uses both mechanisms to provide both robustness and efficiency.

[7] Usually *rtt: round trip time estimate of TCP*
round_trip_time$_{sample}$*: round trip time last sampled by TCP*
rtt-dev: deviation of the roundtrip time, observed by TCP
*rtt + 4*rtt-dev* is used as *rto*

[8] Usually $rtt(n) = (1-\alpha)*rtt(n-1) + \alpha * round\_trip\_time_{samplee}$

complicated Internet today, packets belonging to the same TCP connection can take different routes through the network and arrive at the receiver at a different order than they were sent. The reordering would cause the TCP receiver to send duplicate acknowledgements, just like lost packets. The difference between the two phenomena is the number of duplicate ACKs they cause. In case of out of order packets there would be only a couple of duplicate ACKs, since typically the reordering is not too large. However in case of a packet loss there would be as many duplicate acknowledgements as the window size. TCP accepts three duplicate ACKs as packet loss, and anything less as packet reordering. This provides a reasonably low false alarm probability, but in turn gives rise to other problems. It delays the retransmitted packets by some small amount, due to the detection process, and brings the requirement that there should be at least four packets in flight. If there are less than four packets in flight, the receiver won't be able to generate enough duplicate ACKs to trigger the *fast retransmit* mechanism. This is a problem with low data rate-low delay connections, but in case of the HDR system the bandwidth delay product is well above four packets if the TCP connection always has data to send, so there is no such concern. The fast retransmit mechanism is coupled with *fast recovery*, which adjusts the values of the congestion window size and *ssthresh,* so that the packet flow is uninterrupted other than the lost packet.

2.2 Challenges in the HDR wireless system

On existing wireless links various problems and inefficiencies were observed with TCP. Inherently the inefficiencies are caused by the different nature of a wireless link. The first problem that the wireless data users came across was a dysfunctional fast retransmit mechanism, due to low data rates. This problem does not apply to a system such as HDR that

---

and $rtt\_dev(n) = (1-\beta)*rtt\_dev(n-1) + \beta*(abs(round\_trip\_time_{sample} - rtt(n-1))$

is designed to operate at high data rates. Another problem is the error characteristics of the wireless link. The error rate of a typical wireless link is an order of magnitude larger than a typical wire link and the errors sometimes come in bursts. This means more packet losses for TCP and more packet losses in bursts means more retransmission timeouts for TCP. This is due to the fact that if TCP loses all the packets in a window, it assumes that there is congestion in the network. It is a valid assumption on a wire network, but for a wireless link it is possible for all the packets in a burst get lost due to changing channel conditions, without the presence of any congestion. In such a case TCP unnecessarily goes into congestion control although the bandwidth of the connection did not change at all, and the packet losses were just a temporary problem. The sender underestimates the available bandwidth, because it assumes that a lost group of packets is a sign of congestion. Since this happens once in a while in a wireless link the TCP performance is affected negatively.

There has been a significant amount of work done on the inefficiencies of running TCP over existing wireless systems and many different solutions has been proposed to the problems stated above. To remedy more than usual packet losses, there is the TCP SACK option, which has become standard on most TCP implementation. This simply lets acknowledgements give information about out of order packets received, so that the sender can more effectively decide on which packets to send. To be able to fight the high error nature of wireless networks, one has to both increase the reliability of the wireless link and at the same time make sure that TCP knows the difference between a loss due to wireless link and losses due to network congestion. One can choose to achieve these goals at different layers of the system. Many people proposed to use *performance enhancing proxies (PEP)* that sit at the base station and watch all the packets that go by. There are two approaches for this, as well. The first approach is for the proxy to act as an agent for the mobile host and acknowledge the packets as soon as they reach the base station. This requires a very reliable wireless link to make sure that the packets reach their intended destination. It might sound like a good idea, but it conflicts with the famous *end-*

to-end argument. This design principle says that some connection properties, such as error recovery, should be maintained at the higher layers, end-to-end. Terminating the TCP connection is also prone to risks and brings in a whole new set of issues with it.

Another, more conservative approach is the *Snoop Agent.* This PEP caches all the forward link packets and monitors all reverse link ACKs. If it detects a packet loss due to the wireless link, it retransmits the lost packets from its cache, and suppresses the duplicate ACKs so that the TCP sender does not go into congestion control. This both makes the wireless link more reliable and prevents the sender from underestimating the available bandwidth.

One can also provide link reliability at the lowers layers of the network. The *Radio Link Protocol* is implemented as part of the wireless link and does retransmissions of radio link packets. This approach is accepted by many systems such as IS95, CDMA2000, GSM and HDR. The advantage of this approach is to do faster local retransmissions, without modifying the higher layers. The disadvantage of this approach is that, as the packet is transmitted, its delay increases. Varying large delays are not good for a TCP connection, since they would lead to either unnecessary timeouts or timeouts that would take too long. A change to the timer mechanism of TCP is proposed in the *Eifel Algorithm* and involves a different timeout estimation method, which was claimed to work better than the existing formulas.[9][31]

Another proposed scheme is *Explicit Loss Notification (ELN),* which notifies the sender that the packet loss is due to wireless link loss and not congestion. The counterpart of this is the *Explicit Congestion Notification (ECN),* which informs the sender that there is network congestion emerging.

[9] One clear problem with the timeout calculation is that when the delay gets reduced by a large amount, the rtt-dev increases and the *rto* increases even further. The Eiffel algorithm fixes this issue.

2.3 The HDR system overview

We now describe some important details of the HDR system officially known as *CDMA2000 – 1xEV-DO* [10].[52] The HDR system is designed to prevent as many of the issues above as possible, and achieve high throughput.

2.3.1 System Overview

HDR itself is designed as a layered protocol, following the same design principle as the internet itself. The lowest layer within HDR is the *physical layer* that takes care of transmitting information bits across the wireless link. Just above the physical layer is the *MAC* (Medium access) layer. The purpose of the MAC layer is to control the access to the resources of the physical layer. The *security layer* is designed for future security mechanisms. The *connection layer* is responsible for establishing and managing connections. The *session layer* controls the sessions of the mobile user. And the highest of all the HDR layers is the *application layer* with the two applications: *signaling link protocol* and *radio link protocol.* The first of the two offers fragmentation mechanisms and provides reliable delivery of various signaling messages. The RLP layer provides flow control and semi-reliable transmission for data applications. It provides a byte stream for all higher layer applications. Above HDR we still have the IP protocol suite running and the two architectures are connected by the PPP protocol acting as the *data link* layer, which is a universal protocol that negotiates and manages higher layer protocols running on various point-to-point links.

[10] The reader can always look at the published standards for full detail.

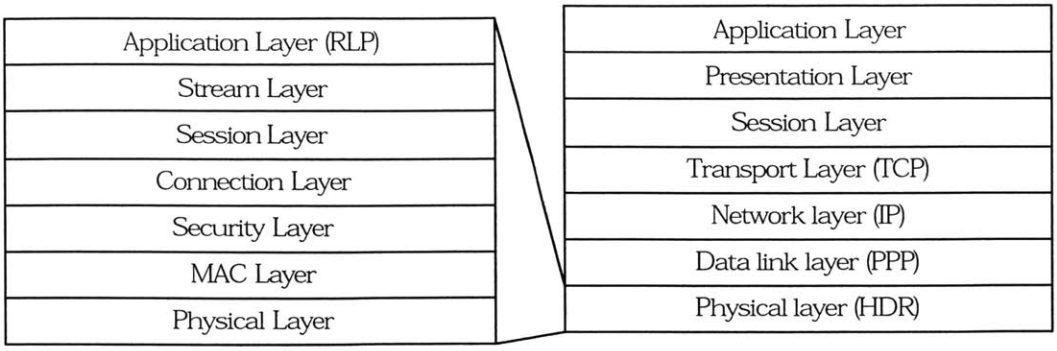


Figure 2.3-1: Different layers in the HDR-Internet[11]

2.3.2 The Architecture

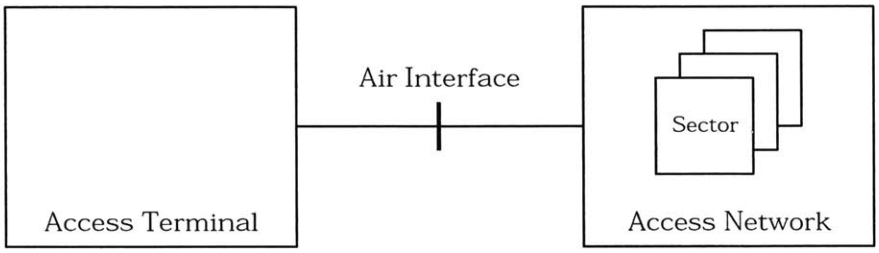


Figure 2.3-2: Architecture Reference Model

The architecture of HDR is similar to the architecture of a wireless voice network. In a cellular voice network when someone wants to make a phone call to a landline phone, the circuit goes through the base station, to the public switch telephone network. In this case the cellular system is an extension of the telephone network. Similarly HDR is designed to be an extension of the Internet. The user has an *access terminal* as a device instead of a cell phone, that functions similar to a modem that attaches to the *terminal equipment.* The HDR infrastructure as a whole is called the *access network*. The interface between the two is called the *air interface*. The access network itself runs all the necessary servers such as DHCP and DNS to be able to get integrated with the Internet. The counterpart of a Base Station Controller in the HDR system is the *Modem Pool Controller (MPC)* and the counterpart of a Base Station Transceiver is the *Modem Pool Transceiver (MPT).* Each sector in the access network can

include a number of HDR channels.[12] Each of these channels require a MPT which is responsible for the transmitting and receiving the traffic for access terminals, and implement the radio link, physical layer and the MAC protocol. The traffic itself is generated at the MPC. The MPC is responsible for maintaining most state information for the connections of access terminals. It acts as the interface between the Internet and the MPTs. The IP packets get queued and get assembled into HDR packets at the MPC and get sent to the MPT over an IP link. The configuration of these components is very flexible. An access network can consist of many MPCs and MPTs, each MPT can be connected to many MPCs and each MPC can be connected to many MPTs. The network configuration becomes important when calculating delays and system performance, specifically in handoff situations. The IP packets arrive at the MPC and the RLP packets get generated and sent to the appropriate MPT that the access point is covered by. This requires both the MPCs and MPTs have queues of packets. Between the MPC and the MPT the connection is done at the RLP level, though the communication between them itself is implemented as an IP network.

[11] This figure and all most other figures in this section are from the HDR standard. [52]
[12] Any number of 1.25 MHz CDMA channels in Network can be allocated for HDR use.

2.3.3 The Physical Layer

The HDR system is TDMA based on the forward link and CDMA based on the reverse link.[13] This design choice has a number of significant performance implications. Since the entire wireless bandwidth is used for a single user at a time, much higher instantaneous data rates can be achieved on the forward link for the served user.[14] This means that in web browsing type application the user perception would be greatly improved. The minimum time unit for HDR is a *slot* which is 1.667 ms long. The transmission of a physical layer packet can take anywhere from 16 slots for the lowest data rate to 1 slot for the higher data rates. I should also mention that a reverse link frame takes 26.666 seconds[15] and this will be useful for delay calculations. Table 1 and Table 2 show the important parameters of the different data rates on the forward link. We can see that the number of slots gets reduced as the data rate increases. Also Table 3 shows the preamble associated with different type packets. We can see that for lower data rates the preamble is longer, in order to ease successful decoding of the packet. Taking advantage of repetition and strong parallel turbo codes, the lowest data rates can be achieved at down to -10 dB. The other data rates vary with 2-3 dB intervals, and tend to have

[13] A wireless communications book should give an overview of these two schemes, but here I will try to give a very brief overview. There are various ways to share a gives frequency band between users. The oldest is to share the frequency band between users by allocating each user a small portion of the band. This was used in the first analog - wireless networks in the US. The GSM standard in Europe is TDMA based. TDMA stands for Time Division Multiple Access and shares the available frequency between users by allocating each user a time slot within a frequency band. This way the capacity of the system is increased by a factor (depending on the number of users sharing the same frequency). Code Division Multiple Access (CDMA) is used in the United States and in the third generation wireless networks. In this scheme all the users share the entire available frequency band. However each user's data is spread by a different code and all the codes are orthogonal to each other. The properties of orthogonality allow the users to decode the correct data that was sent for them. The CDMA system allows more users for a given band size, due to clever coding, power control and system design. HDR uses TDMA on the forward link but at the same time allows the user to transmit over the entire bandwidth.

[14] The reverse link uses CDMA since it increases system capacity and the reverse link could not benefit from a similar TDMA scheme, simply because it lacks the centralization that the forward link takes advantage of, as we will see.

[15] 16 slots

Packet Error Rates (PER) very dependent on the SINR. The shapes of the PER/SINR curves are important in determining system performance.

Table 1: Modulation Parameters for the Forward Traffic Channel and the Control Channel (Part 1 of 2)

Data Rate (kbps)	Number of Values per Physical Layer Packet				
	Slots	Bits	Code Rate	Modulation Type	TDM Chips (Preamble, Pilot, MAC, Data)
38.4	16	1,024	1/5	QPSK	1,024 3,072 4,096 24,576
76.8	8	1,024	1/5	QPSK	512 1,536 2,048 12,288
153.6	4	1,024	1/5	QPSK	256 768 1,024 6,144
307.2	2	1,024	1/5	QPSK	128 384 512 3,072
614.4	1	1,024	1/3	QPSK	64 192 256 1,536

Table 2: Modulation Parameters for the Forward Traffic Channel and the Control Channel (Part 2 of 2)

Data Rate (kbps)	Number of Values per Physical Layer Packet				
	Slots	Bits	Code Rate	Modulation Type	TDM Chips (Preamble, Pilot, MAC, Data)
307.2	4	2,048	1/3	QPSK	128 768 1,024 6,272
614.4	2	2,048	1/3	QPSK	64 384 512 3,136
1,228.8	1	2,048	1/3	QPSK	64 192 256 1,536
921.6	2	3,072	1/3	8-PSK	64 384 512 3,136
1,843.2	1	3,072	1/3	8-PSK	64 192 256 1,536
1,228.8	2	4,096	1/3	16-QAM	64 384 512 3,136
2,457.6	1	4,096	1/3	16-QAM	64 192 256 1,536

Table 3: Preamble Repetition

Data Rate (kbps)	Values per Physical Layer Packet		
	Slots	32-Chip Preamble Sequence Repetitions	Preamble Chips
38.4	16	32	1,024
76.8	8	16	512
153.6	4	8	256
307.2	2	4	128
614.4	1	2	64
307.2	4	4	128
614.4	2	2	64
1,228.8	1	2	64
921.6	2	2	64
1,843.2	1	2	64
1,228.8	2	2	64
2,457.6	1	2	64

The forward link is actually time multiplexed between the *pilot channel, medium access channel (MAC channel), control channel (CC)* and the *traffic channel (*Figure 2.3-3). The pilot channel is used by the ATs as information helpful in synchronizing and demodulating the forward link data. The control channel has the information targeted to all ATs and the MAC channel has AT specific data such as reverse power control bits. The MAC channel and the pilot are just take a constant number of chips out of every slot (Figure 2.3-4). The Control channel is a little more complicated, it takes 13.33 ms every 400 ms, in synchronous transmission, and can take more using asynchronous mode. The scheduling of forward data traffic, has to be within the CC constraint. A 16-slot packet cannot start transmission, just two slots before a CC is capsule is scheduled. We will consider these as just overheads, and some additional delay for the forward link packets (2.3.9 and.2.3.10)

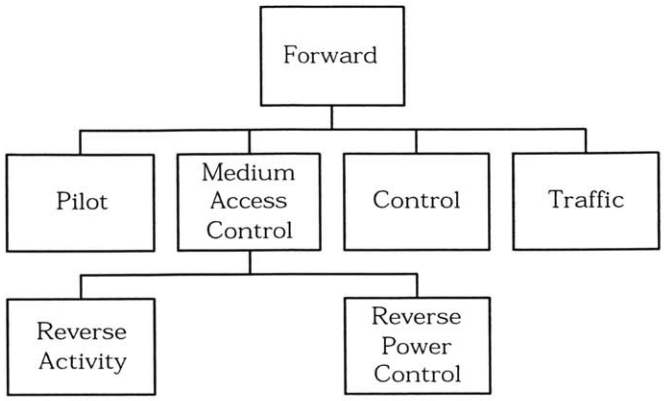


Figure 2.3-3: Forward Channel Structure

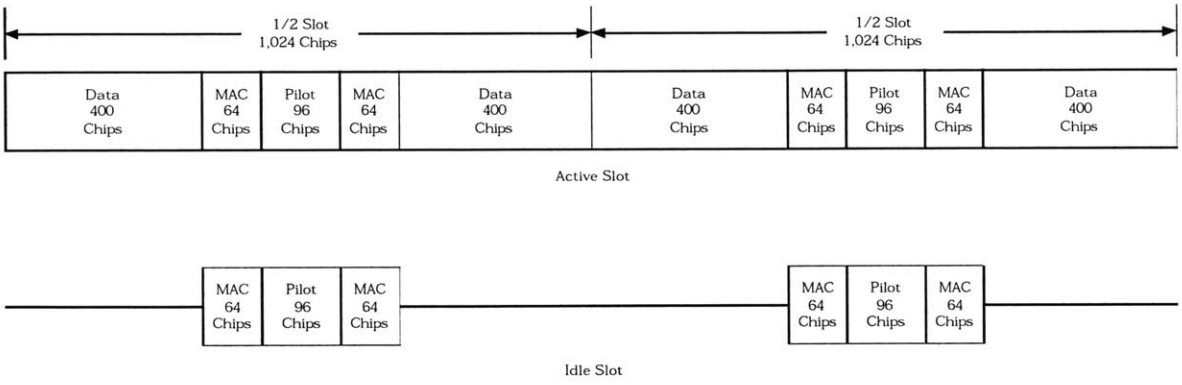


Figure 2.3-4: Forward Link Slot Structure

2.3.4 The Scheduler

Since HDR is a TDMA based system, there needs to be a method to decide which user to serve at a given time. For this purpose HDR uses a scheduler that decides which user to serve. There are two conflicting goals of the scheduler and this results in an inherent trade off. One goal of the scheduler is to be fair in allocating the system resources to different users; another

goal is to get as much sector throughput as possible for the system. One can be extremely unfair and serve the user with the best channel condition, thus achieving a very high throughput. On the other extreme one can make sure that each user gets the same data rate by serving users with low data rates for longer periods of time and serving the user that can support the highest data rate for the shortest amount of time. Another approach would be to use a very simple round robin scheme and serve the users at a predetermined order. This would result in a throughput and fairness somewhere in between the first two extremes. There is no need to get too fair since there is no universal definition for fairness. Even if we agreed on a single definition, one could be unfair over a short period of time and still maintain long-term fairness. The scheduler in HDR achieves exactly this. It keeps track of average data rate that a given user was served in the past and serves the user with the highest *current_rate/averageserved_rate* ratio. This is called a *proportional fair* scheduler since every user gets throughput *proportional* to its average data rate[50][51]. On the other hand for short amounts of time the system can be unfair to a given user. The extent of time that a user will not get any service depends on the time constant of the averaging filter used in calculating the *average_served_rate*. This way each user is served at its peak channel condition and the overall sector throughput is significantly improved compared to a simple round robin scheme. In summary the scheduling algorithm influences the served data rate distribution by increasing the probability of higher rates, and the delay distribution by introducing additional variation.

2.3.5 The DRC mechanism

The essential assumption for the scheduler explained in 2.3.4 to work well is that the scheduler knows what data rate each user will see in the near future. This is achieved by the use of a proprietary predictor and the *data rate request* mechanism. At every slot each user predicts the C/I of the signal it will receive in the future and requests a data rate from the *access*

network.[16] The request uses a dedicated channel in the reverse link called the *Data Rate Control Channel (DRC)*. The prediction of the future channel conditions and the decision of which data rate to request, is essential to be able to get a good performance out of the scheduler. The better informed the scheduler is about the user's conditions, the better decisions it can make. Unfortunately, the prediction and rate request mechanism is quite complex due to both varying channel conditions and availability of many different types of physical layer packets. The data request procedure is not defined in the HDR standard, but it makes sense to implement a good one for good system performance. We will not go into how this works, but will just take into account how it affects the scheduler performance. The data rate requested essentially needs to be conservative for system performance, since sending at a high data rate, when the signal to noise ratio is low would result in unsuccessful transmission of the packet. To minimize wasting slots by unsuccessful transmissions, the access terminal should be confident enough that the data rate its requesting will have a low enough PER at the SINR it is likely to see. On the other hand by being too conservative, the access terminal might not receive the maximum data rate it could support.

2.3.6 The Physical Layer ARQ Mechanism

The scheduler tries to achieve the best possible performance by serving the users when they are in better than average conditions using the DRC values requested that are supposed to track the channel conditions fairly well. This was a major problem in other wireless data services, since they could only support a very low data rate in order to achieve reasonable error rate at the physical layer. The advantage that HDR has over these systems is that it has different packet formats and coding schemes that can perform in various channel C/I values, and it uses very short packet transmissions. These two together with the DRC mechanism enable the HDR system to be able to have more granularity in time, so that the data rate can

[16] Access network is the name given to the fixed side of the HDR system and include the base stations and

change more often thus tracking the channel conditions more closely. This way the excess C/I that the older systems could not benefit from, are utilized.

Still this does not mean that the HDR system can fully utilize the channel. It still has a finite number of data rates and its predictions are not ideal. To recover from conservative predictions there is a *physical layer ARQ* mechanism in the system. This allows packets that would normally take more than one slot to be terminated early in cases where the mobile user succeeds in decoding the packet with fewer slots than specified in the standard. This is possible since a 16-slot packet is essentially the retransmission of the same packet in 16 slots so that the user has a better chance to decode the packet. To be able to implement the physical layer ARQ scheme there is a dedicated channel that carries only one bit signifying if the packet decoding was successful or not. This way the effective number of data rates the system uses is increased. However, it takes time for the mobile terminal to try to decode the packet and the ARQ bit to be transmitted to the BTS so this method requires staggered transmission in the forward link. The amount of time it takes for the access network to get feedback on the ARQ takes 3 slots, this means that there needs to be 3 slot spacing between successive transmissions of a packet to a given user. Consequently the system needs to serve 4 packets at a time, by time division multiplexing the four packets on the forward link. This complicates the scheduler and increases the amount of time the predictor needs to look forward in to the future. Figure 2.3-5 shows the DRC timing for gated transmissions and Figure 2.3-6 and Figure 2.3-7 show a multi-slot packet with normal termination and early termination respectively.

---

the base station controllers of the system.

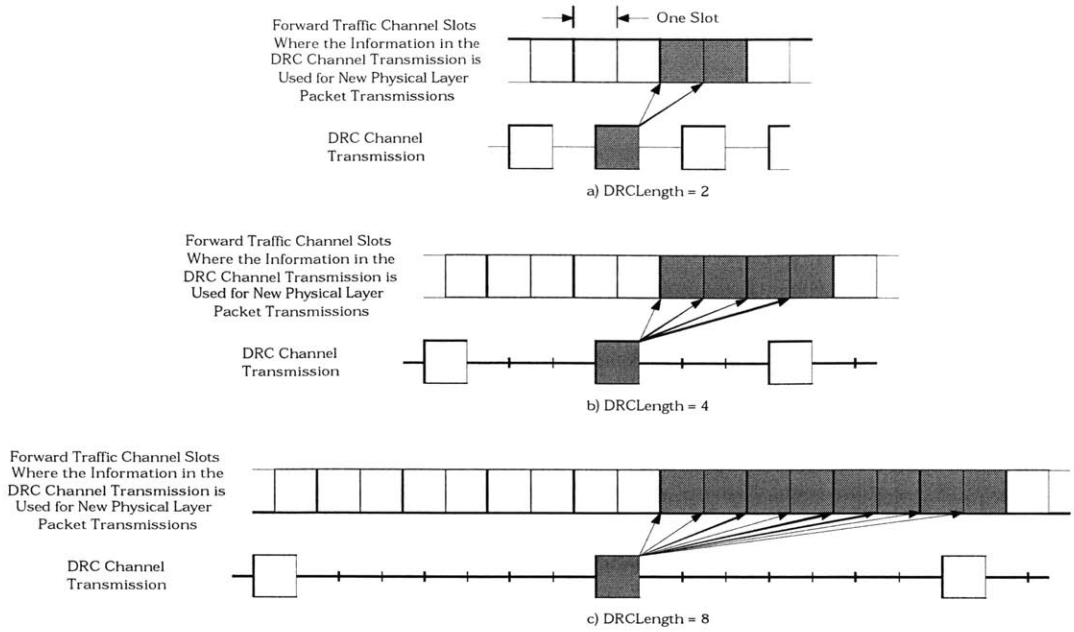


Figure 2.3-5: DRC timing for gated transmission

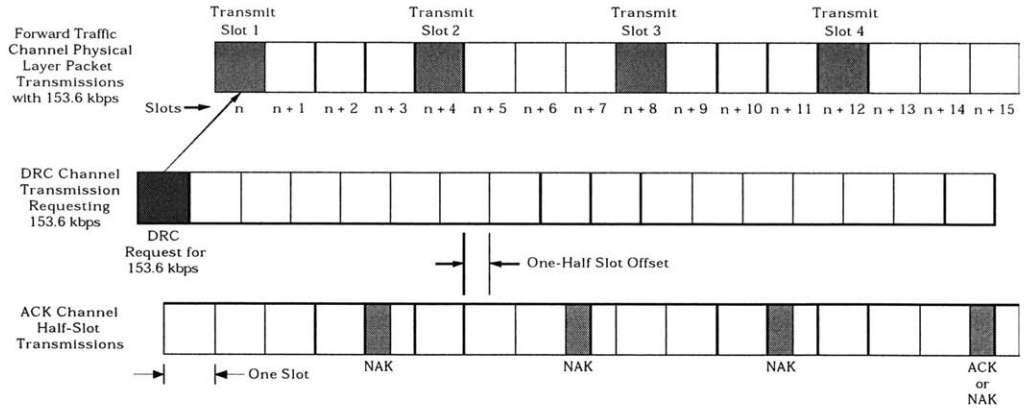


Figure 2.3-6: Multi-slot Physical Layer Packet with Normal Termination

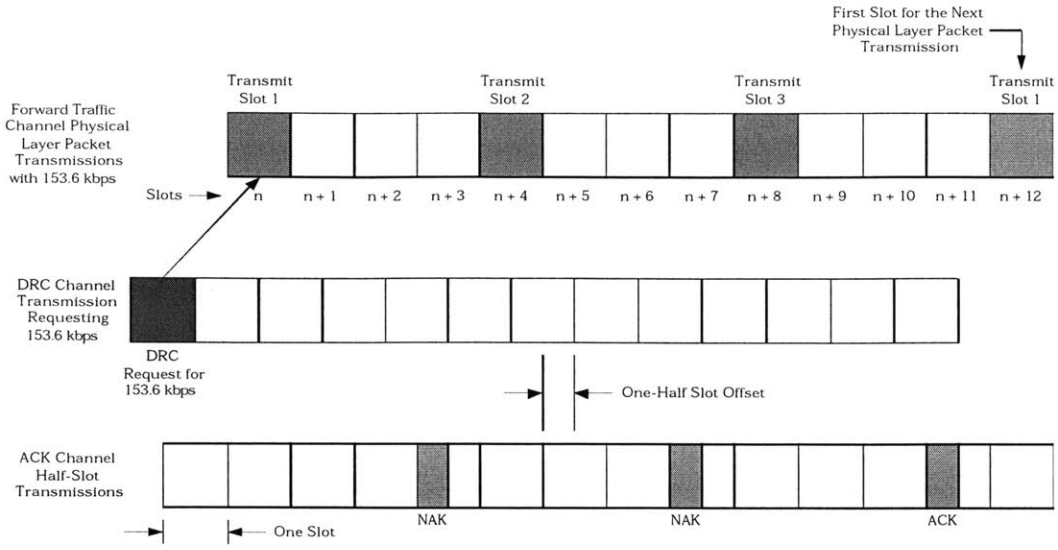


Figure 2.3-7: Multi-slot Physical Layer Packet with Early Termination

2.3.7 The Radio Link Protocol

The last essential component of the HDR system is the RLP layer. The PER of the physical layer is much less than what the transport layer protocols of the internet are designed for. The RLP layer provides single retransmission of packets when necessary by use of *negative acknowledgements (NAK).* A negative acknowledgement acknowledges that a particular *byte (octet)* is missing. The physical layer packet error rate is usually about 1-2%, so single retransmissions were considered to provide good enough reliability to the upper layers. The PPP packets are treated as a byte stream by the RLP layer and fragmented into RLP packets of 1024 bits (Figure 2.3-8). These are than put into physical layer packets and transmitted over the wireless link.

The RLP layer is essentially a semi-reliable transport layer. It provides in-order delivery of RLP packets, retransmission, duplicate detection and flow control. The protocol makes use of sequence numbers of 22 bits for the above goals. For any RLP octet sequence number N, the sequence numbers in the range $[N+1, N+2^{S-1}-1]$ are considered to be greater than

N and the sequence numbers in the range $[N-2^{S-1}, N-1]$ are considered to be smaller than N. There are three state variables involving sequence numbers. V(S) is the sequence number of the first octet of the next packet to be sent by the sender (Figure 2.3-9); V(N) is the next octet for sequential delivery and V(R) is the next new octet expected by the receiver. If V(N) and V(R) are equal it means that all the packets were transmitted successfully in order. If V(R) > V(N), it means that there are some packets that are missing and needs to be retransmitted (Figure 2.3-10). The receiver has a resequencing buffer where the octets are held before sending up to the higher layers. The sender also keeps two queues for each access terminal. One of the queues is for the RLP packets to be transmitted for the first time and the other is for packets waiting to be retransmitted. The RLP layer gives higher priority for retransmission packets, over first time packets. A retransmittion packet for terminal A has higher priority than the first time packets of all terminals. This way the extra delay introduced for a retransmitted packet is minimized. Though the two queues are just used as an easy way to implement this prioritization, the access network can be implemented in other ways as long as it agrees with the standard. The session layer takes care of packet consolidation and de-multiplexing according to priorities, but we will not cover it in detail.

The protocol initiates by setting all variables to zero and clearing all buffers and queues. The sender puts the sequence number V(S) in the header of the RLP packet. The *i*th octet in a packet is actually has the sequence number V(S) + i. The receiver increases V(N) and V(R) accordingly when it receives an octet with sequence number X:[17]

> If $X < V(N)$, the octet shall be discarded as a duplicate.
>
> If $V(N) \leq X < V(R)$, and the octet is not already stored in the resequencing buffer, then:
>
> RLP shall store the received octet in the resequencing buffer.

[17] From the HDR standard [52]

If X = $V(N)$, RLP shall pass all contiguous octets in the resequencing buffer, from $V(N)$ upward, to the higher layer, and may remove the passed octets from the resequencing buffer. RLP shall then set $V(N)$ to (LAST+1) where LAST is the sequence number of the last octet passed to the higher layer from the resequencing buffer.

If $V(N) < X < V(R)$, and the octet has already been stored in the resequencing buffer, then the octet shall be discarded as a duplicate.

If X = $V(R)$, then:

If $V(R) = V(N)$, RLP shall increment $V(N)$ and $V(R)$ and shall pass the octet to the higher layer.

If $V(R) \neq V(N)$, RLP shall increment $V(R)$ and shall store the octet in the resequencing buffer.

If $X > V(R)$, then:

RLP shall store the octet in the resequencing buffer.

RLP shall send a Nak message requesting the retransmission of all missing RLP octets from $V(R)$ to X-1, inclusive.

RLP shall set $V(R)$ to X+1.

When the sender receives a Nak, it will put the requested octets in a packet in the retransmit queue of the access terminal. If the requested sequence numbers are greater than V(S), it is a sign of the protocol breaking up, so a reset is requested. If the requested sequence number is valid, but not available any longer, the Nak is simply ignored. The receiver will set a Nak abort timer to a value of $T_{RLPAbort}$ after it send a Nak for a packet. When this timer expires all the octets from V(N) to the next missing octet, or V(R) are send up to the higher layer. This is why the RLP protocol doe not guarantee full reliability. The RLP octets are retransmitted only once and any further recovery is left for the higher layer end to end protocols.

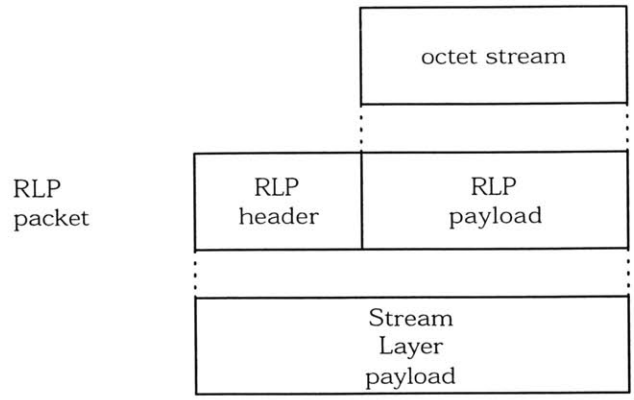


Figure 2.3-8: Default Packet Application Encapsulation

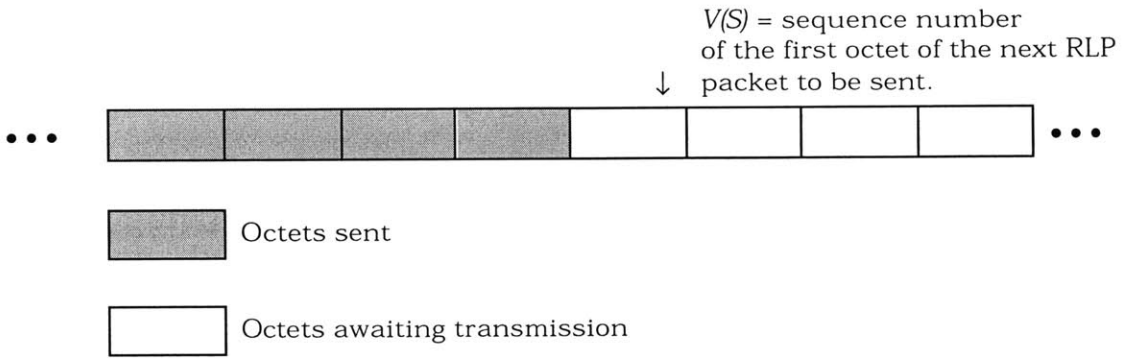


Figure 2.3-9: RLP Transmit Sequence Number Variable

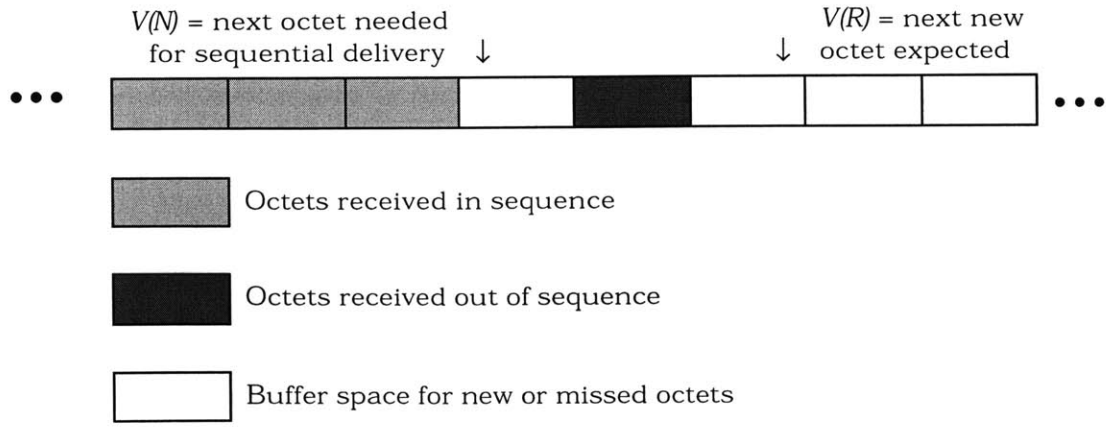


Figure 2.3-10: RLP Receive Sequence Number Variables

2.3.8 Overview

Let's examine the HDR system as an IP packet is transmitted from a server on the Internet to an access terminal. This examination will ignore any protocol initialization and control procedures, but since we are interested in the steady state behavior of connections, it is sufficient. The IP packet reaches the routers at the HDR MPC, and enters the MPC protocol stack (Figure 2.3-1). The PPP protocol at the link layer treats the MPC-AT link like any other physical layer link and forms PPP packets are formed of the IP packet.

The physical layer provides a byte stream service to the logical link layer, and in this case the HDR system as a whole implements the physical layer. This is a good example of recursive layering (Figure 2.3-1). The byte stream is provided by the RLP protocol, which is the highest layer in the HDR system. The RLP protocol is in fact only one of many possible applications of an HDR connection. Since the entire MPC-AT connection is considered to be one physical link, the two sides of this connection are the PPP and RLP end points. The MPC forms RLP packets and sends them to the MPT to get transmitted to the AT. The RLP packets are transmitted along with all other control messages between the MPT and MPC using a backhaul with IP protocol.

Once the RLP packets reach the MPT they go into the first time queue of the corresponding AT. They wait there until the scheduling algorithm decides to serve the AT (2.3.4). This decision is based on the DRC values requested by all the ATs (2.3.5). When the time comes the physical layer packet is formed from the RLP packets. At this point the data rate of the transmission is determined by either the DRC value, or was already fixed due to some other process in the connection. The transmission of the physical layer packet at the given data rate might take a couple of slots. If for some reason the data rate requested was lower than what could have been supported by the channel conditions of the AT, the AT will use the physical layer ARQ to notify the MPT that the packet was successfully decoded in less slots than

planned (2.3.6). This causes the scheduler to immediately serve a new physical layer packet to an AT, saving some slots and increasing the sector throughput (Figure 2.3-7). If not, either the packet is successfully decoded, or the packet is lost. In this case any number of application layer packets within that physical layer packet need to be retransmitted.

The AT can only become aware of the packet loss after it looks at the sequence number of the next RLP packet (2.3.7). The time interval between the lost packet and the next packet is uncertain, since it depends on the scheduling. The scheduler keeps serving ATs, according to the scheduling algorithm, and the time constant of the algorithm determines the longest amount of time an AT could be ignored for. Once the packet loss is detected, a Nak is sent to the receiver on the reverse link. The reverse link is just CDMA, so the Nak will take a fixed amount of time to reach the MPC. The Nak results in a copy of the lost packet to be inserted at the retransmission queue of the AT, at the MPT. This time the retransmission will have higher priority over all other first time transmissions, so the wait for scheduling should be reduced considerably. If the retransmitted packet is decoded successfully by the AT physical layer, it gets appended to the PPP byte stream. From TCP's point of view the packet that was received correctly by the arrival of these lets bytes, only with some extra delay (2.1). This mechanism is good since it reduces the PER of the link to a reasonable value, allowing TCP to safely assume that any loss of packets are due to congestion of the network (2.2). The extra packet loss in the wireless link is replaced by introduction of varying end-to-end delays. In case the retransmitted packet is lost, the AT RLP protocol, just passes whatever it received so far up to PPP, this will result in a packet loss at the TCP level, but is tolerable since it is rare. The error recovery scheme at the RLP level is for performance improvement only, and is not sufficient by itself.

2.3.9 Delays in the System

Now that we have a good understanding of the HDR system, we can calculate the delay that the packets see. This exercise will provide us with fundamental constants of the system and

help us understand how the system works.

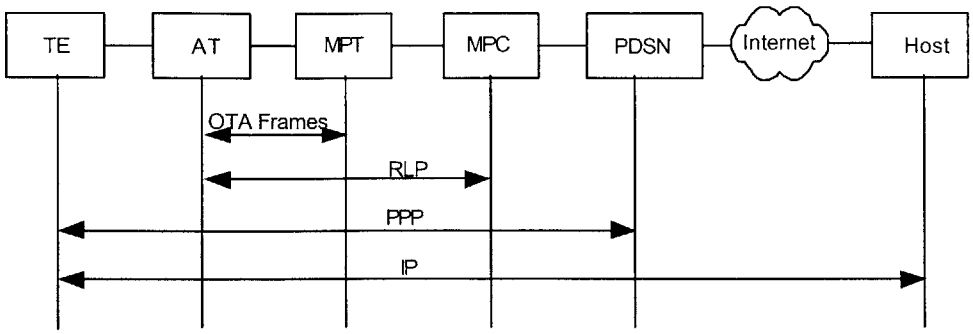


Figure 2.3-11: HDR System Delay Reference Model

We will use the model in Figure 2.3-11. Let's define some delays on this model. The *forward link physical layer delay* is defined as the time from when a HDR forward link packet is scheduled for transmission at the MPT till the packet is received at the AT. The *reverse link physical layer delay* is defined as the time from when a HDR reverse link packet is available at the AT for transmission till the time when it is received at the MPT. The *MPC-MPT transport delay* is defined as the time from when a HDR forward link packet (RLP packet or of a signaling message) is available at the MPC till the packet is received at the MPT. The components of the forward link physical layer delay are listed in the table below.

Table 4: Components of the Forward Link Physical Layer Delay

System Component	Delay Component	Range (ms)	Avg. (ms)	Description
MPT	Packet processing (encoding and modulation)	0.2	0.2	Time spent in encoding and modulation of the packet. Depends on the packet size, hardware speed
MPT	Transmission	Rate dependent (1.67-26.67)	4.17	Time taken to transmit the packet over the air. Depends on the data rate. Table 1and Table 2
AT	Packet processing (demodulation and decoding)	2	2	Time spent in processing the packet, i.e. demodulation and decoding. Depends on the packet size, hardware speed.
MPT, AT	**Forward link physical layer delay**	**4-29**	**6**	**From the time a forward link packet is scheduled for transmission at the MPT till the packet is available at the AT after demodulation and decoding.**

Table 5: Components of the MPC-MPT Transport Delay

System Component	Delay Component	Range (ms)	Avg. (ms)	Description
MPC	MPC-MPT transport processing	0.5-2	1.25	Time spent in MPC-MPT transport protocol stack processing, i.e. UDP (TCP)/IP/Ethernet processing.
MPC	Queuing and transmission at the sender	0.5-2	1.25	Delay in queuing and transmission over the network. (Ethernet delay)
Backhaul	Delay over the backhaul network	1-5	3	Delay in carrying traffic out of the MPC to the MPT. (backhaul and router)
MPT	Queuing at the receiver	0.5-1	0.75	Delay in the MPT receive buffer waiting to be processed.
MPT	Receiver processing	0.5-2	1.25	Time spent in the receiving protocol stack processing to extract the encapsulated packet.
MPC, MPT	**MPC-MPT transport delay**	**3-12**	**8**	**Time to send a unit of payload (RLP packets or signaling messages) from the MPC to the MPT to be transmitted over the air.**

Table 6: Components of HDR System Forward Link Delay

System Component	Delay Component	Range (ms)	Avg (ms)	Description
MPC	GRE extraction and PPP processing	0.5-2	1.25	Time spent in transport processing and extracting the GRE payload before processing by RLP.
MPC	RLP processing and HDR frame construction	0.5-2	1.25	Time spent in forming required RLP packets.
MPC, MPT	MPC-MPT transport Delay (Table 4)	3-12	8	Time to send a unit of payload (RLP packets in this case) from the MPC to the MPT to be transmitted over the air.
MPT	Forward link scheduling delay	0-26.67	2	Time the packet has to wait before it is scheduled for transmission on the forward link.
MPT	Waiting due to Control Channel transmission	0-13.33	0.2	Time the packet has to wait due to control channel transmission. Depends on the CC duration and periodicity.
MPT, AT	Forward link physical layer delay (Table 4)	4-29	6	From the time a forward link packet is available at the MPT till the packet is available at the AT after demodulation and detection.
AT	RLP processing	0.5-2	1.25	Time to process the RLP packet.
AT	Queuing and transmission of the PPP octets to the TE	0.5-2	1.25	Time to queue and transmit the packet to the TE.
HDR system (Relay Models)	**Forward link traffic delay for the HDR system in the relay model configuration.**	**9-89**	**21**	**Delay for a PPP octet to traverse in the forward direction through the HDR system in the relay model configuration.**

The forward link delay in Table 6 is in a way for the best case, with a very small scheduling delay. This would correspond to a high priority packet such as retransmissions, getting scheduled the next available slot. Typically if there are many ATs keeping their queues full, this delay would be large, long enough to cycle through all ATs, but in a short transaction style communication, Table 6 could apply. The reverse link physical delay is very similar to the forward link, except that the number of slots a packet takes is constant (16 slots = 26.66 ms), so the number is a bit larger and constant (Table 8). The total reverse link delay has again this difference and does not have any wait due to CC capsules. These models are called a *relay model*, calculating the delay as a byte stream sees it. Another option would be to use a *network model* when calculating the end-to-end delay and include delay an IP packet sees, but we treat the HDR as a physical link of the network, so the relay model is appropriate.

Table 7: Components of the Reverse Link Physical Layer Delay

System Component	Delay Component	Range (ms)	Avg. (ms)	Description
AT	Waiting for transmission to begin.	0-26.67	13.33	Waiting for the next frame when the transmission can begin. This assumes that there is no queuing and no other transmission is in progress.
AT	Packet processing (encoding and modulation)	7	7	Time spent in encoding and modulation of the packet.
AT	Transmission	26.66	26.66	Time taken to transmit the packet over the air.
MPT	Packet processing (demodulation and decoding)	5	5	Time spent in processing the packet, i.e. demodulation and decoding.
AT, MPT	**Reverse link physical layer delay**	**38-65**	**52**	**From the time a reverse link packet is available for transmission at the AT till the packet is available at the MPT after demodulation and detection.**

Table 8: Components of HDR System Reverse Link Delay in the Relay Model Configuration

Logical Component	Delay Component	Range (ms)	Avg (ms)	Description
AT	PPP buffer delay	0.5-1	0.75	Time spent in the buffer waiting to be processed by RLP.
AT	RLP and HDR frame processing	0.5-2	1.25	Time spent in forming required number of RLP packets and subsequently the HDR frames.
AT, MPT	Reverse link physical layer delay (Table 7)	38-65	52	From the time a reverse link packet is available for transmission at the AT till the packet is available at the MPT after demodulation and detection.
MPT, MPC	MPT-MPC transport delay (Table 5)	3-12	8	Time to send a packet received over the air at the MPT to the MPC.
MPC	RLP processing	0.5-2	1.25	RLP processing delay at the MPC
MPC	Queuing and transmission of the PPP octets to the PDSN	0.5-2	1.25	Time spent in encapsulating the octets, queuing and transmission over to the PDSN.
System (Relay Model)	**Reverse link traffic delay for the HDR system in the relay model configuration.**	**43-84**	**65**	**Delay for a PPP octet to traverse in the reverse direction through the HDR system in the relay model configuration.**

2.3.10 Overheads in the System

As we can see there are many overheads the system. Let's make an account of where these come from in Table 9.

Table 9: Overheads in the system

Overhead	Explanation	Value
TCP layer overhead	20 bytes/packet (1500 bytes)	1.3% loss
IP layer overhead	20 bytes/packet (1500 bytes)	1.3% loss
PPP layer overhead	8 bytes/PPP frame(1508 bytes)	0.53% loss
HDR frame headers	6/128 bytes	4.6875% loss
HDR frame loss	Physical layer PER	2% (current target)
Overhead due to CC	13.33 ms out of 400 ms	3.3325% loss

2.3.11 Throughput Ratios for different layers of the system

In this section we can look at the throughput at different layers in Figure 2.3-12 and verify that their ratios agree with the overhead calculations from section 2.3.10.

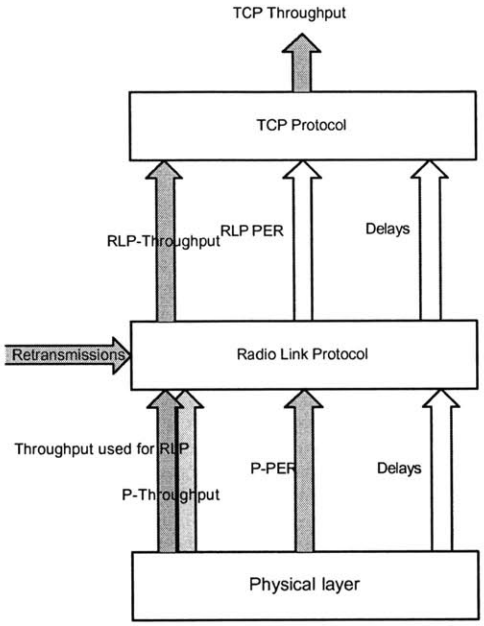


Figure 2.3-12: Overview of the layer throughputs analyzed

Figure 2.3-13 shows the ratio of the TCP throughput to the RLP throughput that includes all RLP transmissions including lost packets. The throughput loss between the two layers is due to the header overheads and TCP's congestion control mechanism. This value should be:

3% (headers) + Physical layer PER + Loss due to TCP window back-offs.

Figure 2.3-14 shows the ratio of the throughput at the RLP layer to the throughput at the physical layer used for RLP. This should be approximately equal to the *(1 – physical layer PER)*, since the RLP throughput is equal to the Physical layer goodput. In Figure 2.3-15 the effect of control channel slots, show up as 3% for low error rates and get higher as the forward link utilization goes down due to TCP back-off. A full implementation of the system would

also include signaling packets, which would reduce this ratio. It is interesting to note that only Figure 2.3-14 is independent of persistency of the RLP layer. The others all have lower utilization as TCP backs-off due to lost packets. Figure 2.3-16 is basically the product of the first three figures, and shows the utilization a TCP user gets out of the HDR physical link.

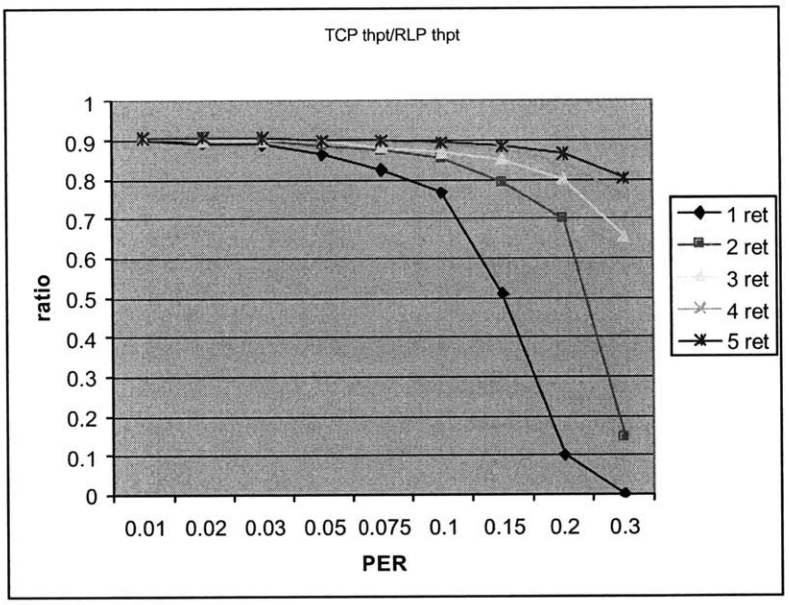


Figure 2.3-13: Ratio of TCP throughput to RLP throughput

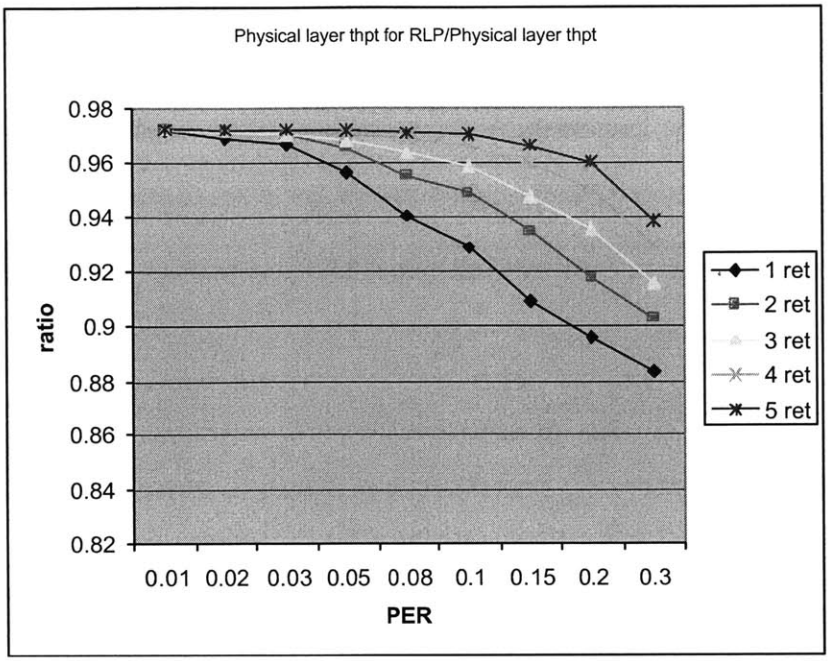


Figure 2.3-14: Ratio of RLP layer throughput to Physical Layer throughput

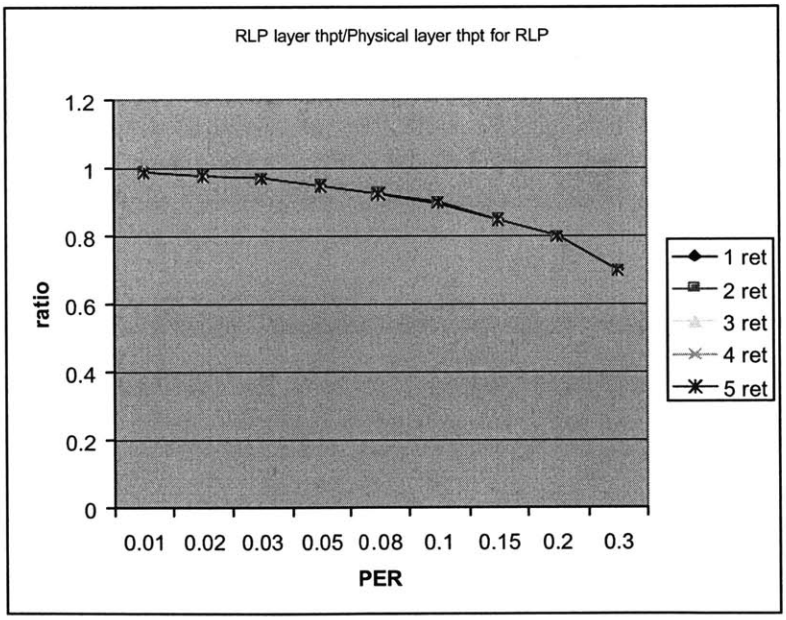


Figure 2.3-15: Physical layer throughput used for RLP/Physical layer throughput

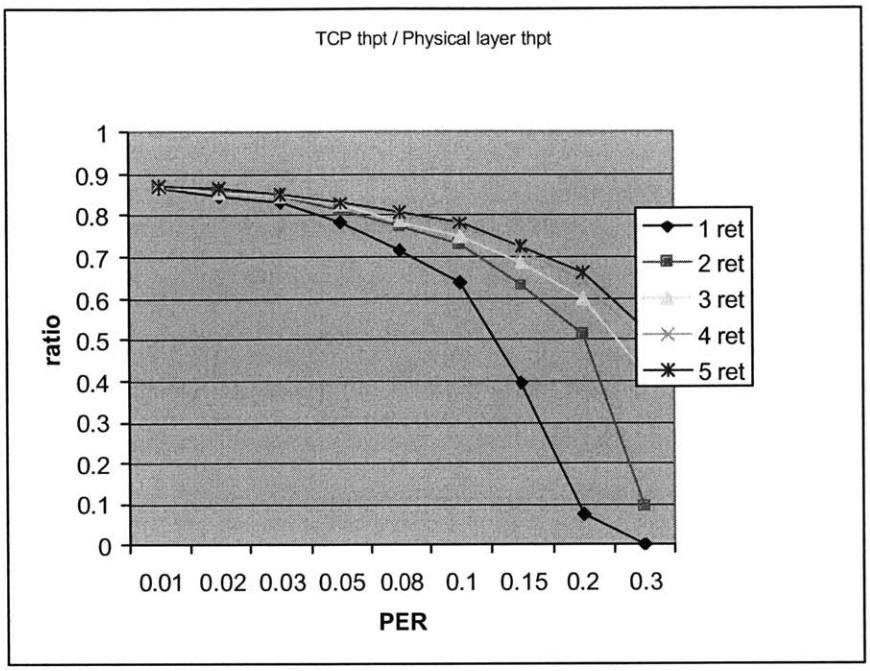


Figure 2.3-16: Ratio of TCP throughput to Physical Layer throughput

Chapter 3: The Simulation Environment

The network level simulation of the system was done using *OPNET[56]*. OPNET is a commercial simulation tool often used in similar projects. There is also a free tool available named *ns*, but OPNET was chosen due to the availability of earlier development in this platform. OPNET is mainly an event driven simulation, and has a friendly graphical user interface. The user can construct the network out of existing components or create new ones. In this case to be able to speed up research, as many of the standard components as possible were used. All the available tools were taken advantage of, and the simulations are very functional, though correct. OPNET was used for simulating RLP layer and higher layers. The physical layer was, simplified and simulated by OPNET, simulated by other c-based simulators, or logged from real life systems. Figure 2.3-1 is an overview of the full simulation, showing the data flows between different components and adjustable parameters of each stage of the simulation. The details of the simulations are not required in order to understand the results, since the simulation only implements the HDR system and TCP/IP protocol, already explained in Chapter 2. This section can be skipped with the assumption that the simulations are functionally correct.

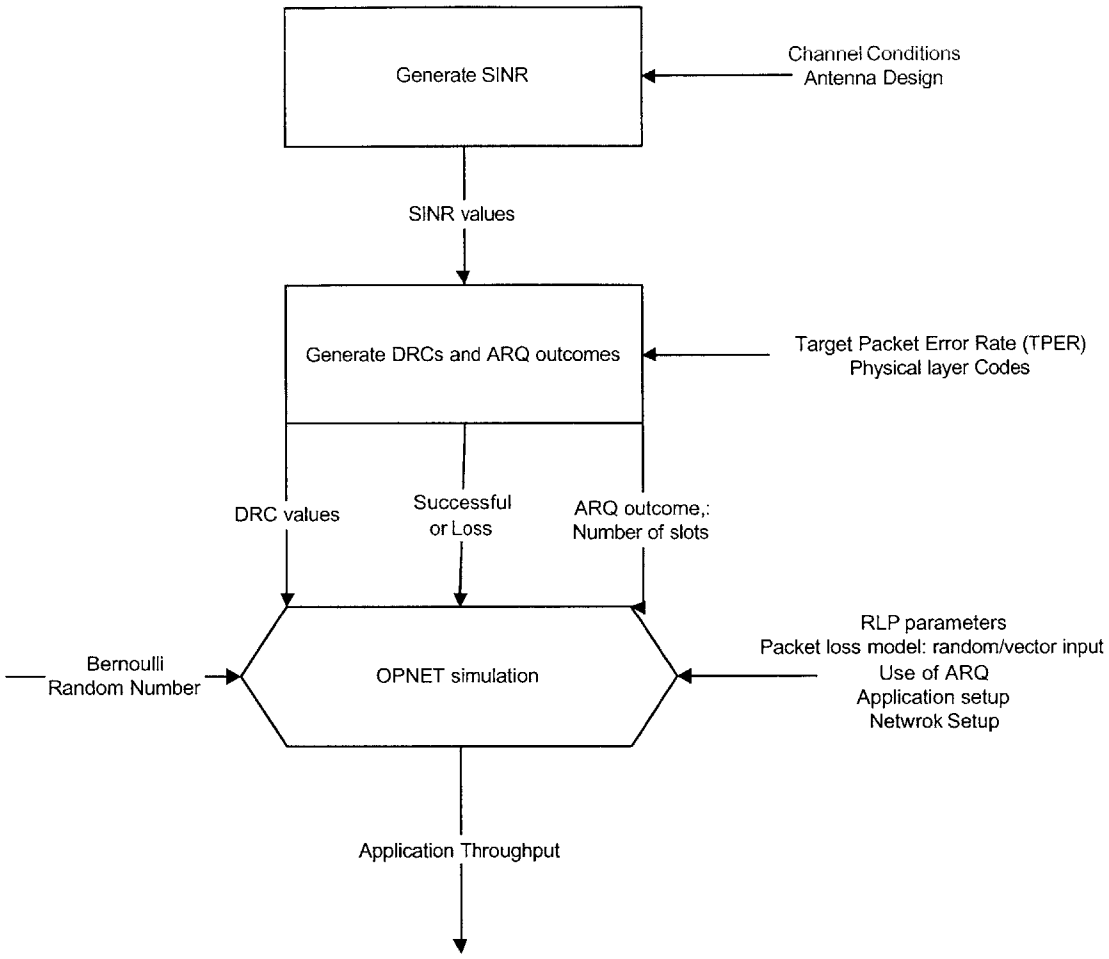


Figure 2.3-1: Simulation Overview

3.1 OPNET simulation

The Opnet network model is shown in Figure 3.1-1 and corresponds to the HDR architecture as in section 2.3.2. The simulation model relies on built in models and processes, for everything above the OSI's physical layer, implemented in the Servers and the Workstations. The servers are connected to an IP cloud, which is connected to the MPC. The MPC is connected to an MPT and the MPT is connected to a special node that represents a set

of 64 Access Terminals. The Access terminals are connected to workstations through IP links. The MPC-MPT link is a 10Mbps link, the IP cloud and AT64-MPT links are 100bps.

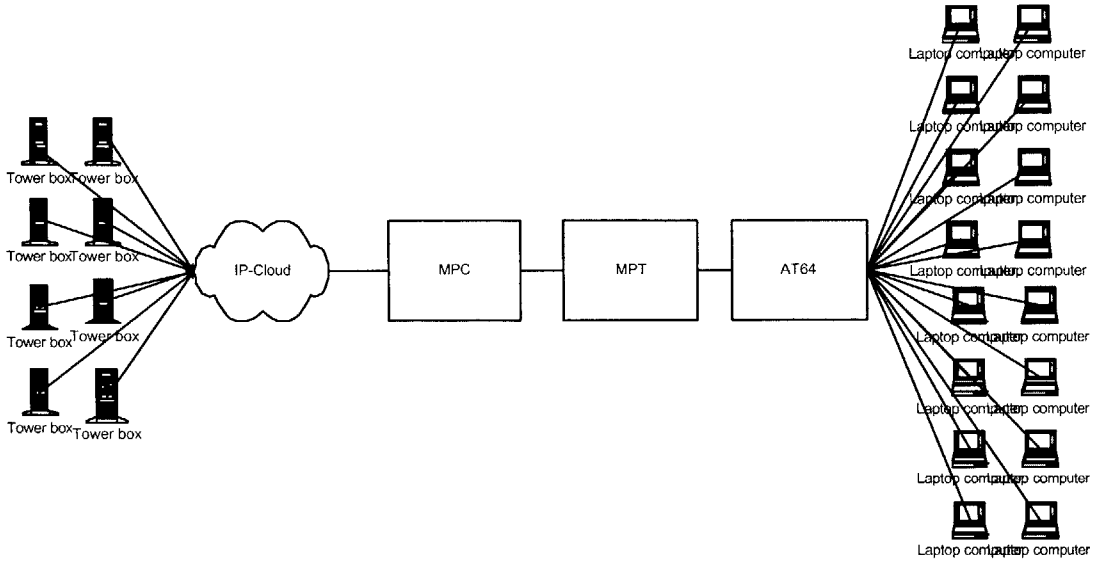


Figure 3.1-1: OPNET Network Model

Figure 3.1-2, Figure 3.1-3, and Figure 3.1-4 show the node models for the MPC, MPT and AT64 and are representative of the process-path the packets follow through between the workstations that represent the terminal equipment and the IP-cloud. The Access Network and Access Terminal nodes do not implement the IP protocol for ease of simulation development, so the routing of the packets to the correct workstations and servers is done by multiplexing and de-multiplexing the packets onto the correct links at the AT64 node. AT64 implements the RLP protocol for packets traveling on the reverse link. The whole protocol is implemented as one process, and that drops some packets and adds appropriate delays. The RLP packets are formed from the PPP byte stream and their fate is determined by the outcome of a random number generated by a Bernoulli Process. The error rate is taken as 2% on the reverse link. If the outcome is successful, the packet is delivered to the MPT node with the appropriate delay, otherwise another random number determines if the Nak for the lost packet is successful. If this is also a failure, the packet is discarded, if the Nak is successful, the random number is

generated again for the retransmission, this time the packet is either successful, or dropped since the RLP layer implements only single retransmissions (2.3.7). If the packet is successful after a retransmission, it is penalized by the time it takes for the reverse link RLP receiver to detect the packet loss and one roundtrip time for the Nak and retransmission to propagate. The reverse link packet gets propagated through the MPT and MPC to the IP-cloud, with the average delays added. The reverse link is only implemented in order to complete the loop for TCP, is used only for application level requests by the workstations and TCP feedback by the workstation, so it is not an essential part of the simulation.

The forward link is the main focus and the downstream TCP connection has the server as the sender and the workstation as the receiver. In this case the MPC just forwards the IP packets by adding the necessary overhead bits. Unlike the real access network, the RLP sender of the forward link is not the MPC, but it is the scheduler. So the *RLP Tx process* in the MPC node just adds overhead bits and the RLP functionality is in the *scheduler* process in the MPT. Implementing these two processes together simplified the implementation, though created a more complex process. The Scheduler reads the DRC values of access terminals from an input vector, and decides on which AT to serve. The RLP sender and the receiver is again implemented in one state machine, that accounts the forward link resources a packet takes, and adjusts the delays the packet sees. This process is the core of the OPNET simulation, allowing us to try out various forward link designs and study effects of different mechanisms. This time the number of maximum number of Naks sent by the receiver before the sender gives up on RLP octets and the value for $T_{RLPAbort}$ is adjustable. Also the forward link errors can be read in from the input vector, instead of being generated by a random process, and the packets can all use full transmission, or early termination reading ARQ outcomes from the input vector.

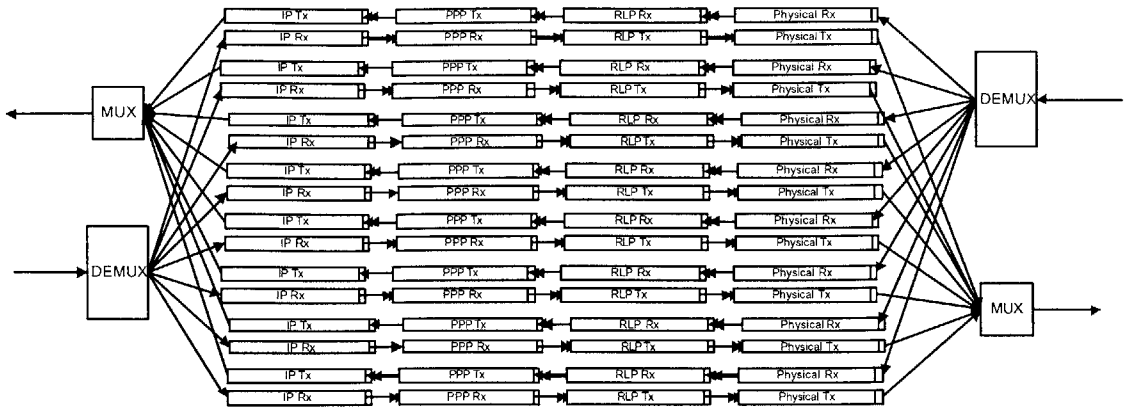


Figure 3.1-2: MPC node model

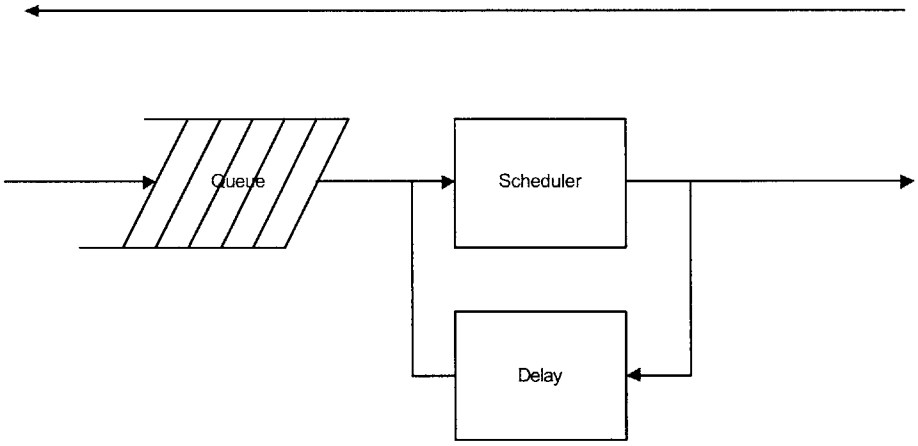


Figure 3.1-3: MPT Node model

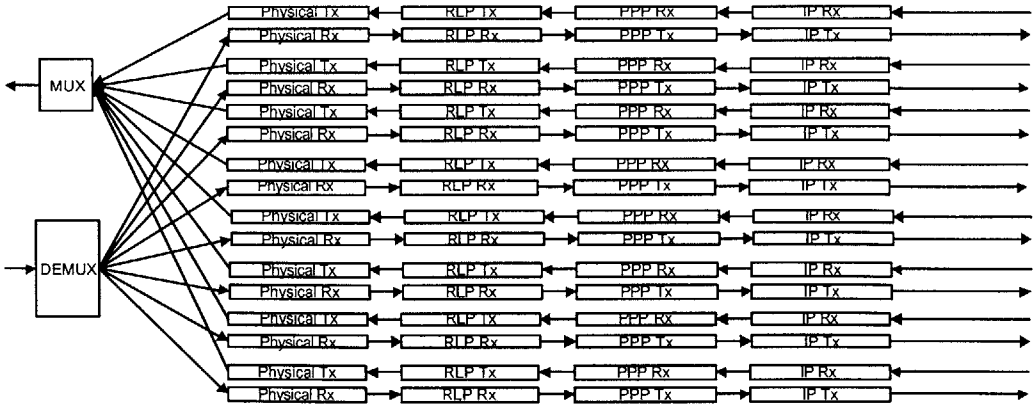


Figure 3.1-4: AT64 Node Model

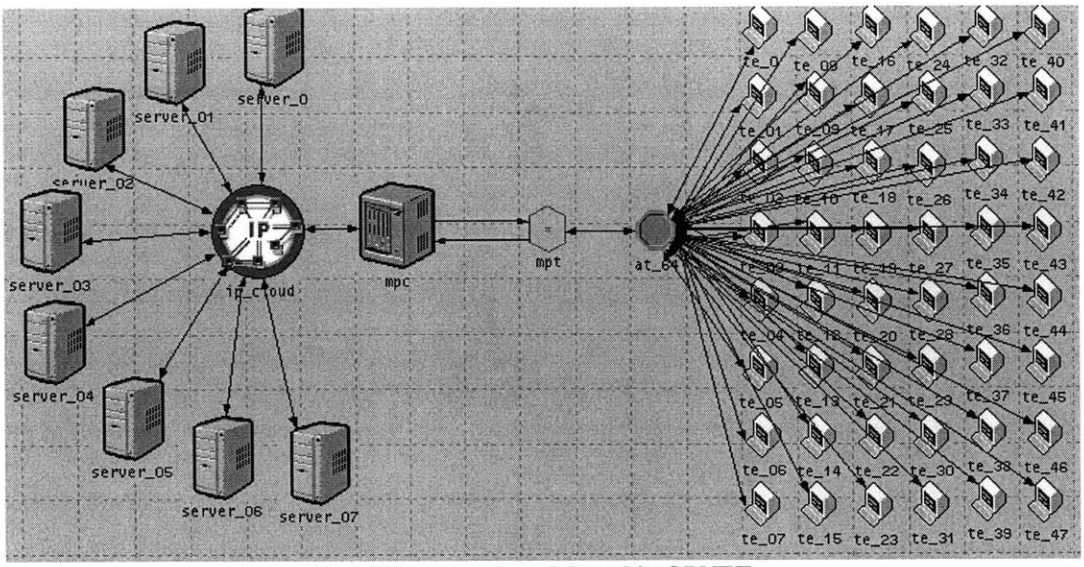


Figure 3.1-5: The general model used in OPNET

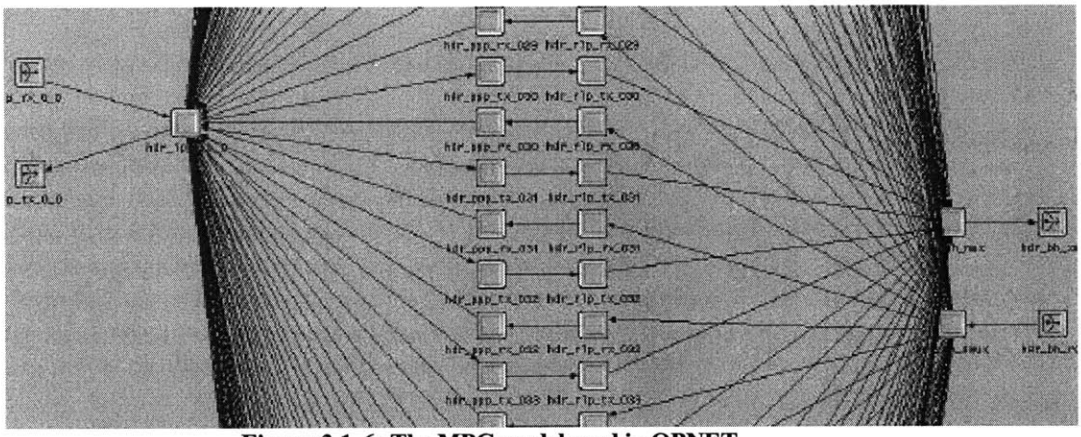


Figure 3.1-6: The MPC model used in OPNET

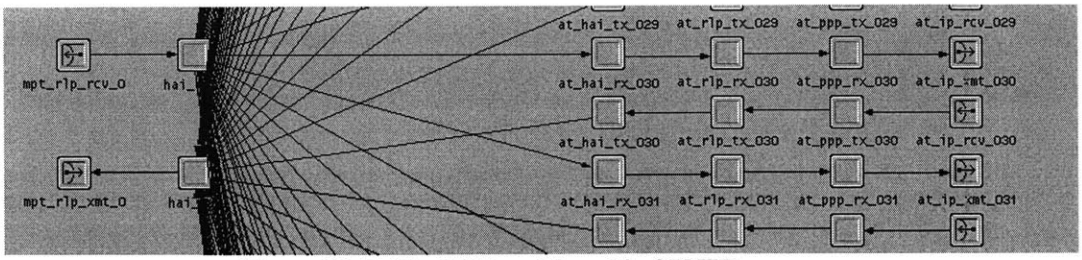


Figure 3.1-7: The MPT model used in OPNET

Various Network setups with varying number of access terminals and different applications were simulated as the different parameters of the system were tweaked. Chapter 4 presents these setups and the results. The OPNET part of the simulation is functionally equivalent to the access higher layers of the access network, with some simplifications. There are no physical Naks sent on the reverse link, and the reverse link is modeled for the forward link RLP connection as having constant delay and error rate. This assumption is acceptable since we are interested in evaluating the forward link performance.

The RLP protocol uses the timer with value $T_{RLPAbort}$, to decide on when to give up on missing octets. This timer determines the time interval between successive negative acknowledgements and the period the RLP receiver will wait before giving up on missing octets. If the timeout value is too small, octets will be discarded even if they are successfully retransmitted. This will inactivate the RLP protocol if it is common enough. On the other hand if the timeout value is too high, RLP will wait for too long, before giving up on lost octets. This time the delay seen by successful octets, just following lost octets, is increased. The effect of this is to boost up the delays TCP sees, and increase the values of *RTT* and *RTO*. This again is not a desired situation, so it is necessary to find a good value for $T_{RLPAbort.}$ OPNET simulations were setup as explained in section 3.1.[18] We observed that the throughput did not change much if $T_{RLPAbort.}$ is between 200ms and 1s and degraded if it is outside this interval, so we chose 500ms as the value of $T_{RLPAbort}$ [19].

There are different ways to improve the RLP layer persistency. The main extension to the single-Nak single-retransmission scheme is to send more Negative Acknowledgements by the receiver when missing octets are detected. In the method we considered, the receiver sends

---

[18]. The network consists of 8 users that connect to 8 different servers through an IP-cloud with 100ms average delay, with 20 ms standard deviation, and no packet loss. It is safe to assume that the servers are not congested, and the access network is the only source of packet loss. The servers and workstations all use windows 2000 implementation of TCP/IP, and the servers send the workstations very large files, using the FTP protocol. The errors on the wireless link are generated by the outcome of a Bernoulli Process, and no early terminations are permitted. Also the simulation duration is chosen as 300ms.

up to *max_rlp_retr* Naks, in regular intervals of $T_{RLPAbort}$. The sender RLP just retransmits the packet every time it receives a Nak.[20] Another approach would be to increment the number of Naks sent after each $T_{RLPAbort}$. In our implementation the Naks follow a (1 1 1 1 1) pattern, a more aggressive approach would be (2 3). The performance difference between these two approaches was studied in [49]. The (1 1 1 1 1) scheme was found to outperform if the errors are correlated, or when the TCP window size is large if the errors are not correlated. The (2 3) scheme is used in the CDMA 2000 standard. The (1 1 1 1 1) method takes longer to recover from errors, and the mean delays are larger, however reduces the unnecessary retransmissions. We did not focus on finding the best pattern, but the literature suggested this would perform better, in the HDR system since the TCP delays and window size is relatively large. Even a more aggressive scheme is to retransmit more than once for each Nak received. In our opinion the exact Nak pattern used in RLP could shift some of the numerical results, however would not effect the conclusions drawn from this thesis. The (1 1 1 1 1) pattern with single retransmissions per Nak was used in all the following simulations.

3.2 Physical layer simulations

The physical layer simulations are a set of c-based programs, and luckily they were fully developed by the time this project started by Qualcomm. Depending on the wireless channel conditions and antenna models a set of SINR values are generated for each access terminal. From the SINR values, a set of DRC values and the results of the transmission at the requested data rates are simulated. The output of this set of simulations is a set of DRC values and the transmission outcomes, which is failure or success and includes number of slots

[19] Same value as in the standard
[20] This parameter is shown as "RLP Retr" on most of these figures.

required in case of an early termination (2.3.6). The DRC request mechanism includes a proprietary predictor of the SINR, and depending on the SINR a DRC value is requested. When the DRC values are requested, a physical layer packet error rate is targeted, which corresponds to the long term steady state physical layer PER that should be observed.

This data is formatted appropriately and fed into the OPNET simulation. The characteristics of the physical layer coding algorithms are built into the simulations. The error characteristics of the coding schemes are assumed to have hard thresholds. The real PER characteristics are as in Figure 3.2-1, however the simulation assumes a failure if the SINR is below a threshold and success if above the threshold and early terminations are decided by, calculating the combined SINR of all the received slots so far. These simplifications are okay, and were verified earlier, to give correct results, by the author of the simulators.

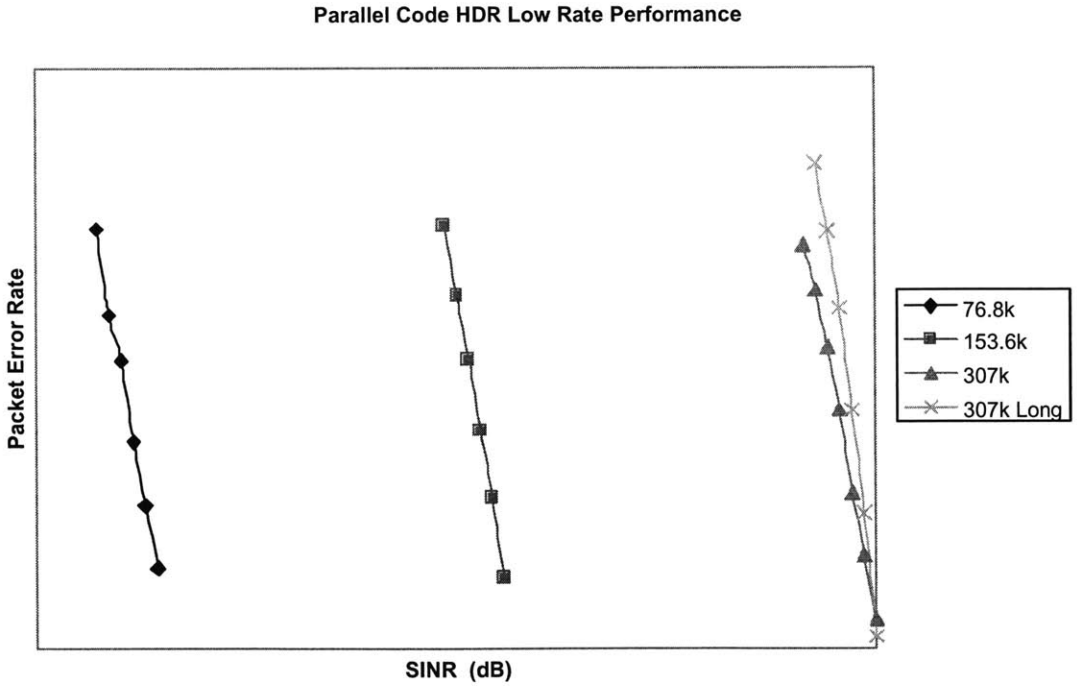


Figure 3.2-1: PER Performance for HDR codes

3.3 Real life data

The Over The Air network present in San Diego was used for getting system logs. The logs were taken for an embedded sector, with up to 16 access terminals. The early state of the OPNET simulation was tested before this project by feeding in the logged data as an input vector, and comparing the OPNET simulation result with the lab results. These tests were successful, however the extended versions of this simulation model could not be tested since, the extended features would take too much time to implement on the physical network. And the ease of development is the main reason for using simulations in the first place. Though the simulations match for the base case of the original HDR standard with the implementation, and the results all have been conceptually understood by more than one person so it is safe to assume that the simulations are as correct as any other available tool.

All levels of the system were logged real time, as the applications were started on the test network. *Chariot* by Ganymede Software was used to start applications with different characteristics, and log their performance between the servers and the terminal equipment, laptops with HDR modems in this case. The TCP layer was logged by running *TCPdump* on both the servers and the clients, this data was fed into *TCP Trace* and plotted by *Xplot.* The HDR level, data was done by Qualcomm's in-house developed logging and analysis software. Again the existence of all these tools greatly accelerated the work. The logged data was used just to analyze real life behavior to look for any unexpected behavior. There weren't any surprises, however it was illustrative to look at the data from all different layers.

Chapter 4: Discussion of Results

In this chapter the results will be presented and discussed. First we study trade off between error control at the RLP layer and the Physical Layer. Then in section 4.2 the findings are discussed using the suggested model.

4.1 Throughput versus Packet Error Rate and RLP Persistency

There are several error recovery schemes in HDR. At the lowest layer the coding used in the physical layer is a type of error control scheme. All the physical-layer packet transmissions use the same chip rate for modulating the base-band signal. However, the different data rates are just different ways of coding, repeating and framing these base-band bits. Naturally, a low data rate takes advantage of codes with stronger error recovery, and uses more repetition. If a packet is transmitted successfully with fewer repetitions than planned, the physical layer ARQ scheme terminates the transmission, avoiding redundant repetitions. This can be thought of as an enhancement to the error recovery scheme at the physical layer. The second error recovery mechanism is at the RLP layer. The RLP layer provides a limited number of retransmissions for missing packets requested by negative acknowledgements.

The error rates that each of these protocols provide to the upper layers can be adjusted. At the physical layer this can be achieved by choosing a different coding scheme for the physical layer packets. However the rate selection at the physical layer is a complicated process. The most influential parameter at the physical layer is the *target packet error rate (TPER)*. TPER determines the average packet error rate the physical layer aims to maintain. At the RLP layer,

the error rate can be adjusted by choosing a different negative acknowledgement policy at the receiver and retransmission policy at the sender.

At the physical layer of HDR, a packet error rate of 0.02 was targeted, mainly due to the intuition of the people designing the system and partly due to the existence of the codes used in voice systems. This error rate is too high for TCP to function efficiently, so at the RLP layer single retransmissions based on single negative acknowledgements were implemented, which lowered the packet error rate TCP sees to 0.001. For good performance of TCP, this is probably the desired error rate. However, the same error rate can be achieved by a different combination of error recovery schemes at the physical and RLP layer.

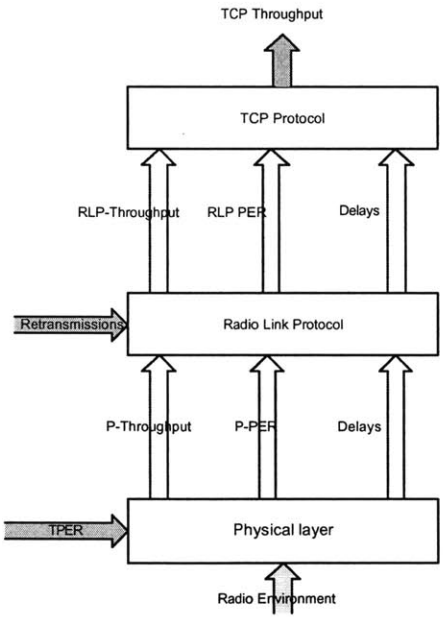


Figure 4.1-1: TCP throughput as a function of TPER and RLP

Figure 4.1-1 summarizes the parameters used in adjusting the persistency of the physical layer and the RLP. Figure 4.1-2 shows the TCP throughput the application layer sees

as these parameters change in a km30 channel.[21] We did not consider TPER below .01 since the improvement due to TCP would be small at these error rates, and the physical layer throughput would have to be reduced significantly to achieve lower error rates at the physical layer. We observe that there is some improvement in the throughput for this channel condition. However, km30 channel exhibits the best improvement, while the other channels show little improvement, and in some cases, performance loss with increased TPER. This system does not have a specific environment for which it is targeted, so we conclude that increasing the TPER at the physical layer is not a good idea.

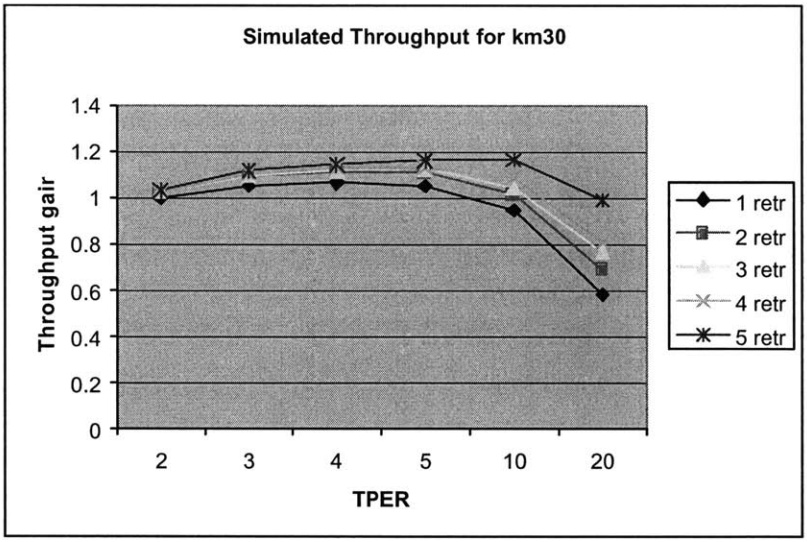


Figure 4.1-2: Simulated throughput gain in km30 channel

Furthermore, these simulations do not take into account the effect of the physical layer ARQ mechanism. Figure 4.1-3 shows the throughput gain due to the physical layer ARQ mechanism and Figure 4.1-4 shows the gain due to increased TPER in the presence of the

[21] The AWGN channel just has Additive White Gaussian Noise, which should be simply random. The Indoor model corresponds to an access terminal in a room with a strong multi-path and is very close to the AWGN model. The km3, km30 and km120 are channel models corresponding to vehicles traveling at 3 kmph, 30 kmph and 120 kmph, having strong fading effects. At km3, the fading is slow enough to trace and at km120 the fading is fast enough that there is time-diversity, but for km30 none of these apply and predicting the SINR is hard.

physical layer ARQ mechanism. We observe that the physical layer ARQ mechanism improves the performance of the system significantly. However the gains we observed earlier due to increased TPER disappear with the physical layer ARQ. The real-life implementations do include this mechanism so we conclude that there is not a significant benefit of reduced error control at the physical layer for this system. We can conclude that for this system 0.02 TPER and single retransmission RLP is a good combination, as long as the physical packet error rate does not exceed 0.05. However we know from experience with real-life systems that the PPER may increase up to 0.10 for short periods of time, so it might make sense to implement a slightly stronger RLP layer with double Nacks. In section 4.2 we will explain the reasons for these observations.

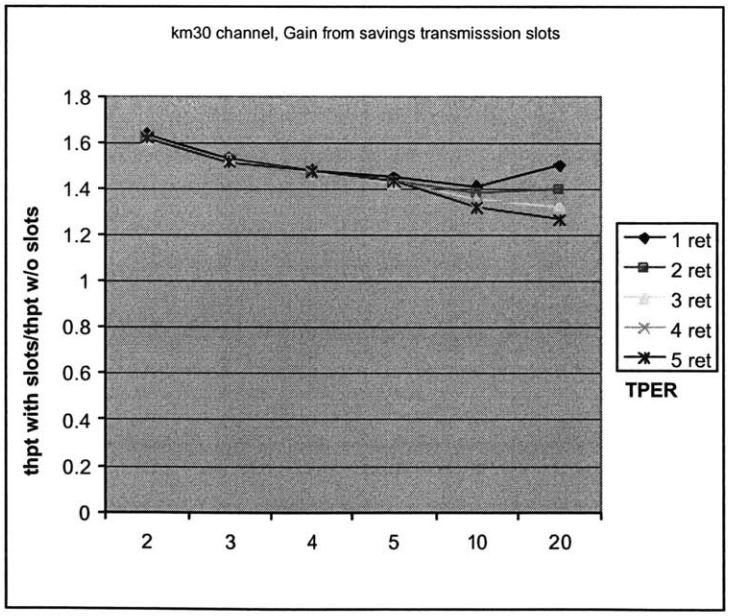


Figure 4.1-3: The throughput gain, by using early termination

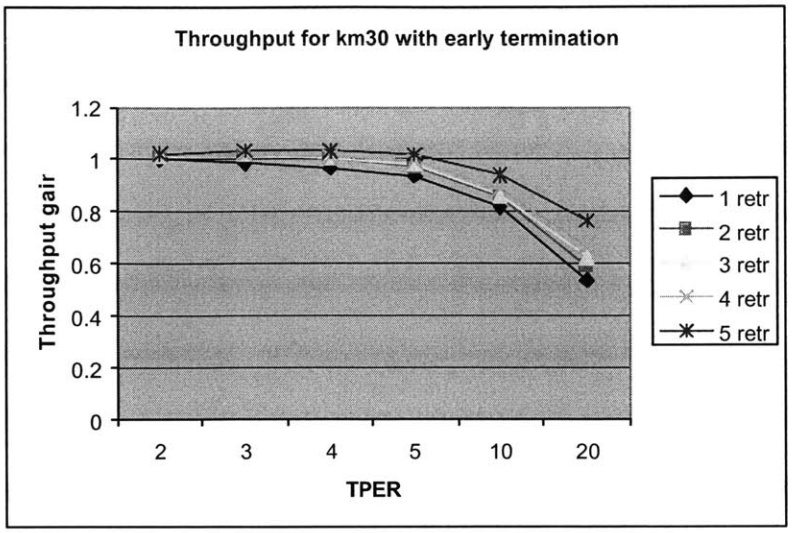


Figure 4.1-4: Throughput gain with TPER for km30 with early termination

4.2 Framework and Empirical Model

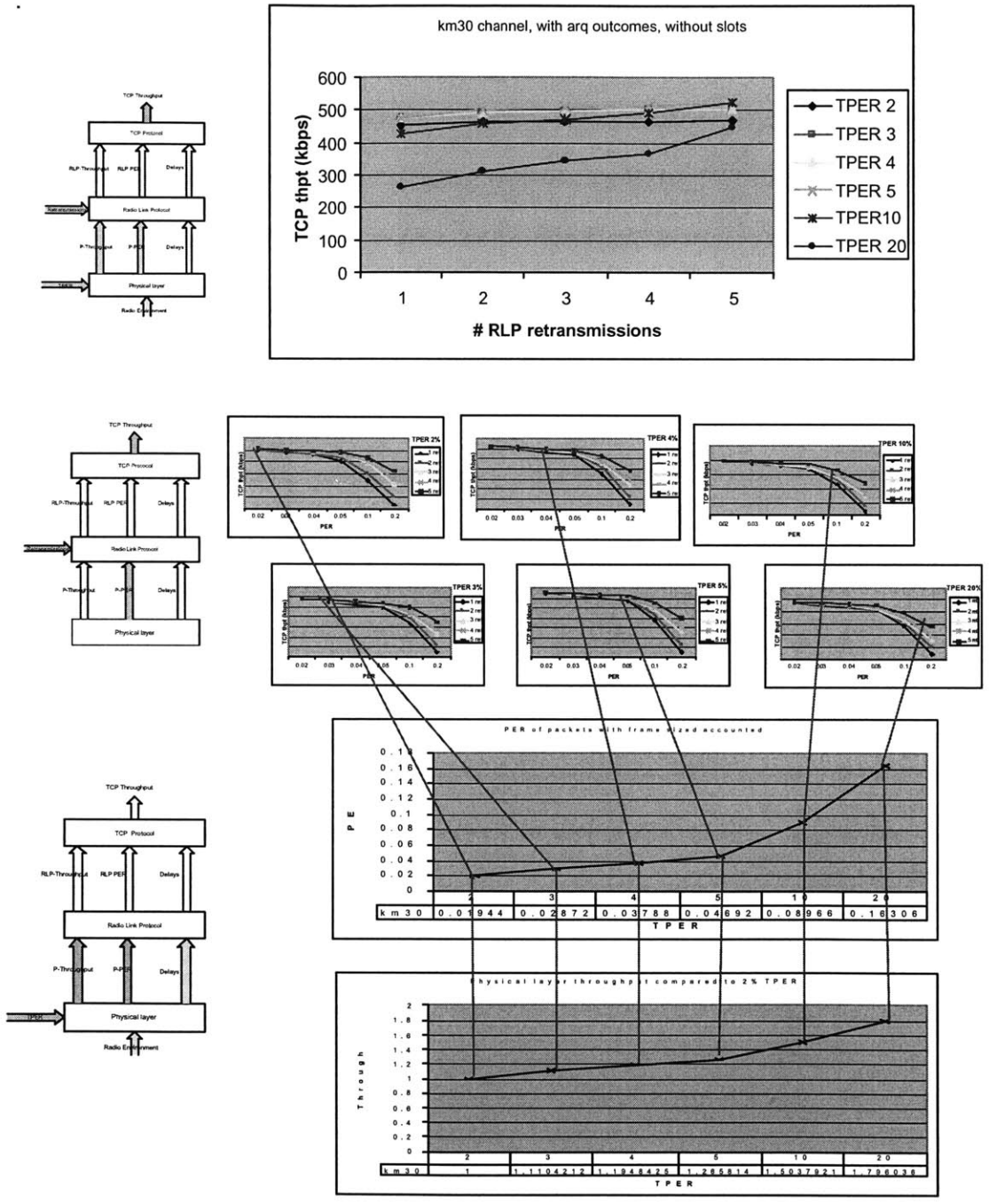
We divided the system into three layers (Physical, RLP, TCP) for analysis and determined the important parameters of each layer. We characterized the physical layer for different channel conditions in terms of packet error rate and throughput and also analyzed the sensitivity of the TCP throughput to physical error rate through simulations. The characterizations of the different layers were used as empirical models, and combined to show that the interactions defined between the layers are reasonably complete. This characterization and model of the system is the second important contribution of this research.

An increase in TPER, causes both P-throughput and PPER to increase. The first of these reduces the TCP throughput, and the second increases it. We can write the TCP throughput as

$$Throughput_{TCP} = Throughput_{Physical}(TPER) \bullet (1 - PPER) \bullet Overhead_{Headers} \bullet Overhead_{TCPcongestionAlgorithm}(PPER, \max\_rlp\_retr)$$

This would mean that the physical layer throughput and PPER influence the TCP performance independently.[22] This can be verified by simulating the entire system and comparing the results with what we would predict using the above formula. Let's look at the km30 channel model and explain the results observed in section 4.1. Figure 4.2-1 shows how this formula can be used to predict the system throughput.

[22] It is actually known that these two are not independent, at least for very low throughput in which one of the link characteristic does not allow the full TCP algorithm to work without constraints.



$$Throughput_{Physical}(TPER) \bullet (1 - PER_{Physical}) \bullet Overhead_{TCPcongestionAlgorithm+headers}(PPER, \max\_rlp\_retr) = Throughput_{TCP}$$

Figure 4.2-1: Breaking up the TCP throughput

4.2.1 Effect of Target Packet Error Rate on Physical Layer Characteristics

The standard leaves the rate request algorithm up to the implementers, but most likely the rate decision will be made based on some kind of prediction of the future channel conditions, with some desired physical layer characteristics in mind.

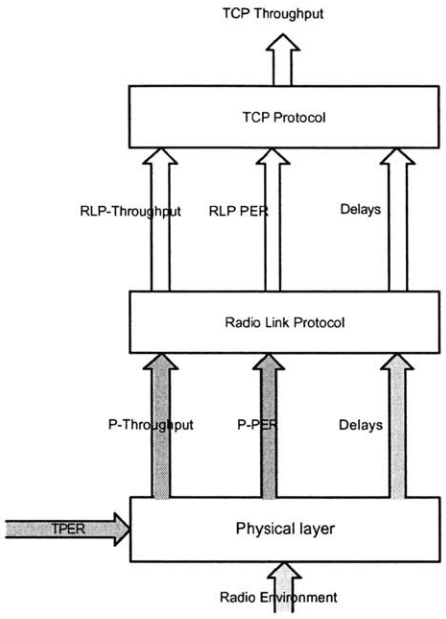


Figure 4.2-2: The mechanisms that influence physical layer characteristics.

In this section we will not go into the details of the algorithm, and will treat this part of the system as a black box. The simulations in this section implement Qualcomm's algorithms, and only use the C-based simulators shown in Figure 2.3-1. Any rate request algorithm can be adjusted to achieve a targeted physical error rate, so we consider this as the essential parameter of the physical layer. Characteristics such as the raw throughput or the delays could be the other design concerns depending on the application, but the packet error rate seems to be a good choice.[23] All these characteristics tend to be very closely related and heavily dependent

[23] HDR is designed for Data applications, and explicitly not for voice, so having stable error characteristics for the link layer is a highly desired property, thus the adjustable knob of the system can be assumed as the PPER for most implementations.

on the channel conditions, so the tests were repeated for a range of simulated wireless environments. The series of SINR values is what we actually get out of the channel conditions, and there has been a lot work done on modeling the channel, but these are out of the scope of this thesis. The environment is essentially not in the system designer's control, but the applications can be targeted to indoor versus outdoor environments, or high-speed vehicles versus stationary stations. Figure 4.2-2 summarizes the relationship we are trying to understand in this section.

Figure 4.2-3 is calculated by simulating the system as the TPER and PER change. The values are the average of the ratios of the TCP throughput as the TPER is changed, while PER is held constant. We can see that there is significant variance due to different channel conditions. The largest physical layer throughput increase is in the 30km case, and the lowest is in the awgn case. Figure 4.2-4 shows the packet error rate and in general the physical layer packet error rates are smaller than the target values in these simulations. This is just a feature of the system, and these relationships can be different for other systems or implementations. However some trends can be generalized and it is important for us to characterize these relationships, to understand the trade-offs.

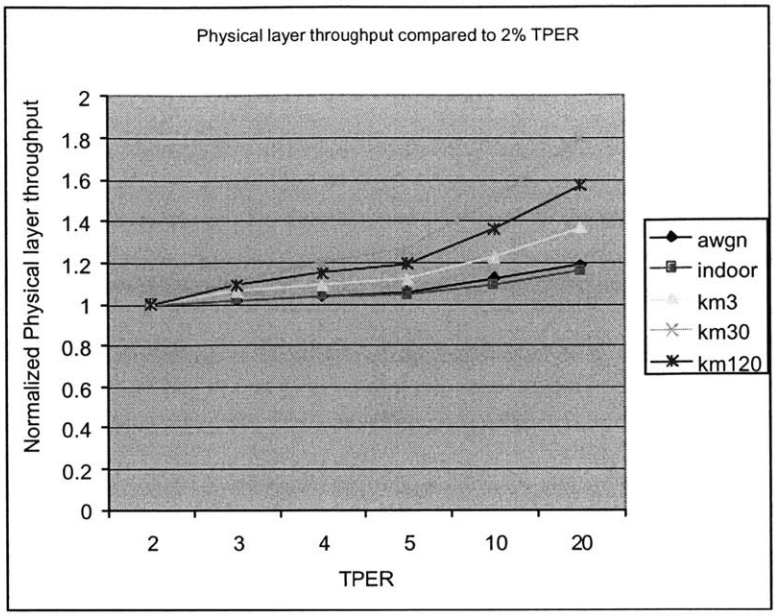


Figure 4.2-3: The increase of the DRC values as the TPER increases.

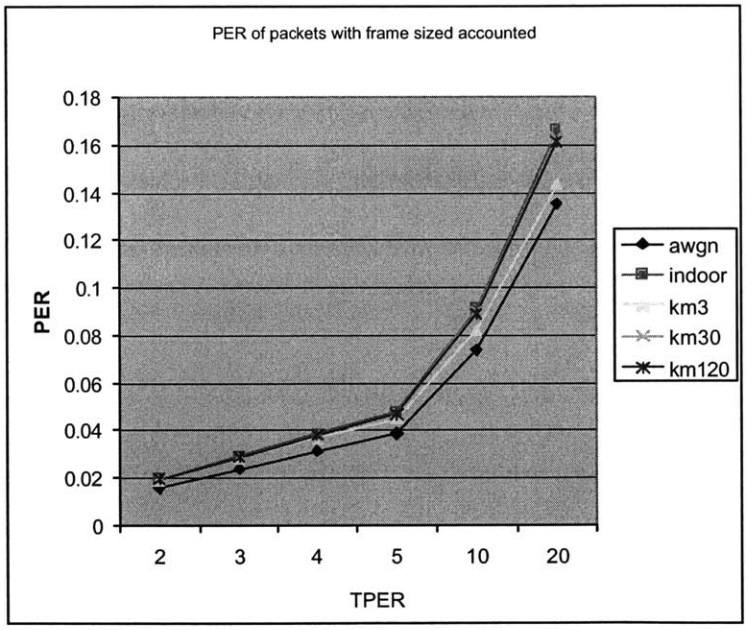


Figure 4.2-4: The PPER as the TPER changes.

We can analyze the errors in order to understand better how the physical layer works. The average packet error rate can be expressed as:

$$PER = \sum_{data\_rates} \left[(\%Slots_{data\_rate}) \bullet (PER_{data\_rate}) \right]$$

Figure 4.2-6, Figure 4.2-7 and Figure 4.2-8 show the percentage of slots for which the data rates increased, decreased, or stayed the same and Figure 4.2-9, Figure 4.2-10, and Figure 4.2-11 show the PER for each category.

Let's go back to how the rate request works and assume that we have infinite data rates we can choose from. For every data rate, there is a PER-SINR relation, so for a given SINR we can look at these relations and find out a PER-data rate relation (Figure 4.2-5).

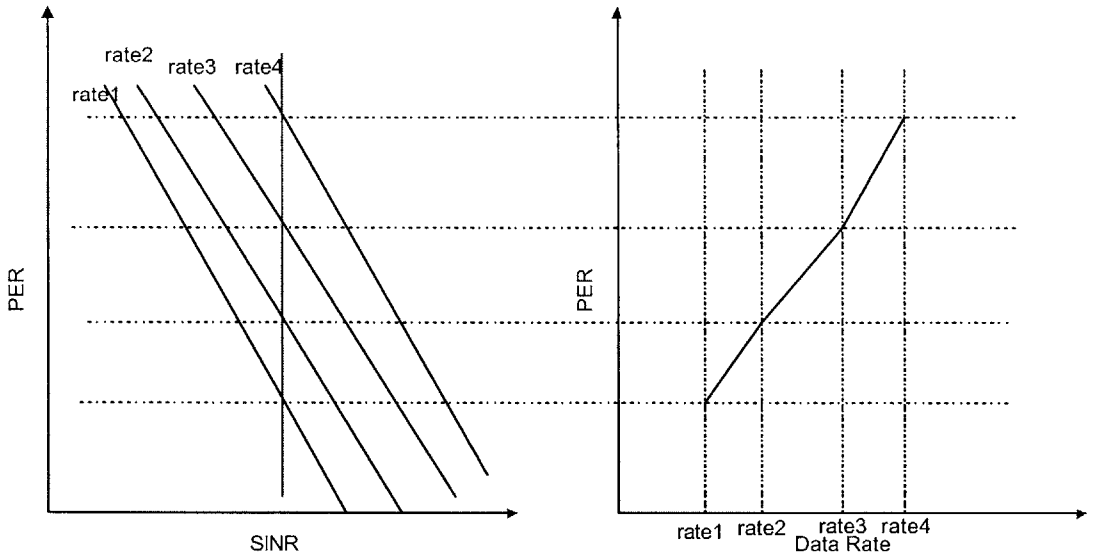


Figure 4.2-5: PER-data rate plot, from a PER-SINR plot

The rate request algorithm inherently needs to be conservative, since errors are undesirable in data applications. When there is a lot of variation or the channel conditions are unpredictable, more conservative data rates need to be requested. When the TPER increases, the new requested data rates are higher, but will still have low error probability. On the other hand, a well-known channel such as AWGN will have a higher PPER to start with, so the increased data rates will have even more probability of error. In all conditions the average PER increases as the TPER is increased, but the mechanism is different in each case due to the channel behavior. In the AWGN-indoor channels, the DRC is not increased for many slots, but

the error probability is high if the DRC is increased, since the original rate requested was already good based on good knowledge of the channel. In the mobile vehicle environments, more slots have increased DRC values, but the PER for the slots with increased DRCs is not as high. In other words, the algorithm is trying to reach a target PER and the equation below holds in all cases:

$$TPER = (PER_{increased} \bullet Percentage_{increased}) + (PER_{same} \bullet Percentage_{same}) + (PER_{decreased} \bullet Percentage_{decreased})$$

The AWGN channel $PER_{increased}$ is higher than $PER_{increased}$ for a high varying and unpredictable channel such as km30. So in order to achieve the same TPER in km30 channel, $Percentage_{increased}$ should be higher, and this means that the increase in average physical layer throughput is higher since the data rates increase more. Equivalently, by increasing TPER, we are allowing the rate selection algorithm to increase some of the data rates, accepting a certain PER. The algorithm gets a higher hit in average PER for every slot with an increased DRC, in the AWGN channel than km30. Thus for the km30 channel, more slots can have their DRCs increased before reaching the same TPER, causing a higher increase in average throughput. The more varying and unpredictable the channel is, the more improvement in throughput gained by increasing TPER:

$$ThptIncrease_{AWGN} \approx ThptIncrease_{Indoor} < ThptIncrease_{km3} < ThptIncrease_{km120} < ThptIncrease_{km30}$$

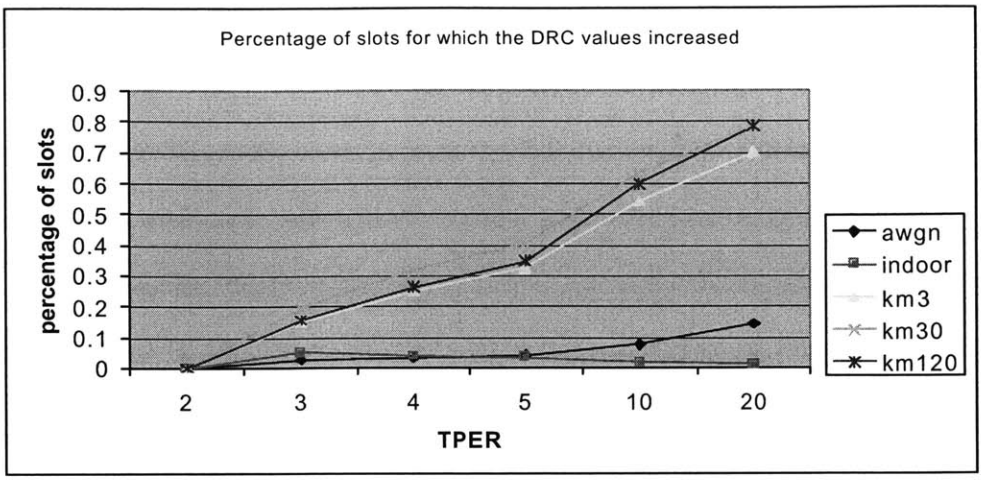


Figure 4.2-6: Percentage of slots for which the DRC increased

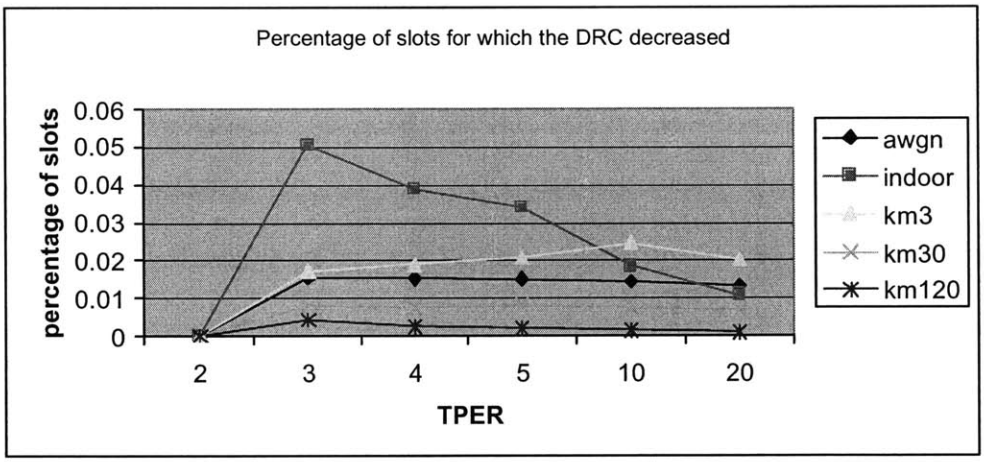


Figure 4.2-7: Percentage of slots for which the DRC decreased

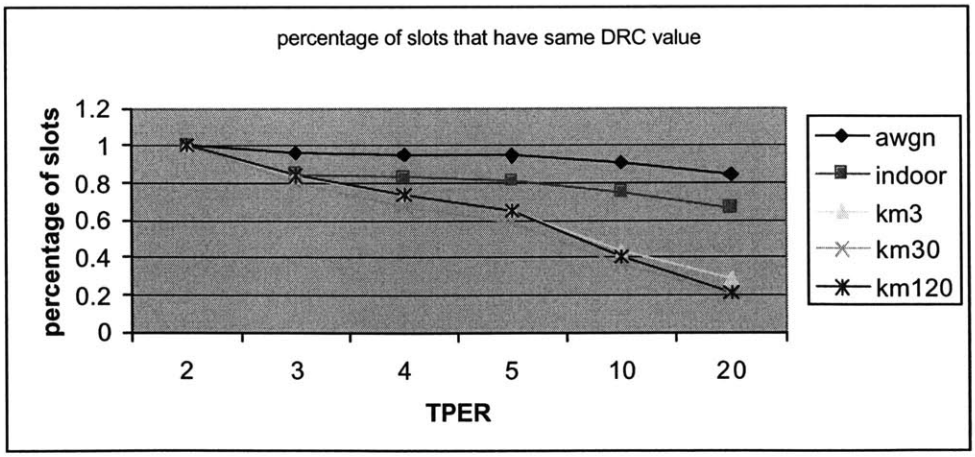


Figure 4.2-8: Percentage of slots for which the DRC stayed the same

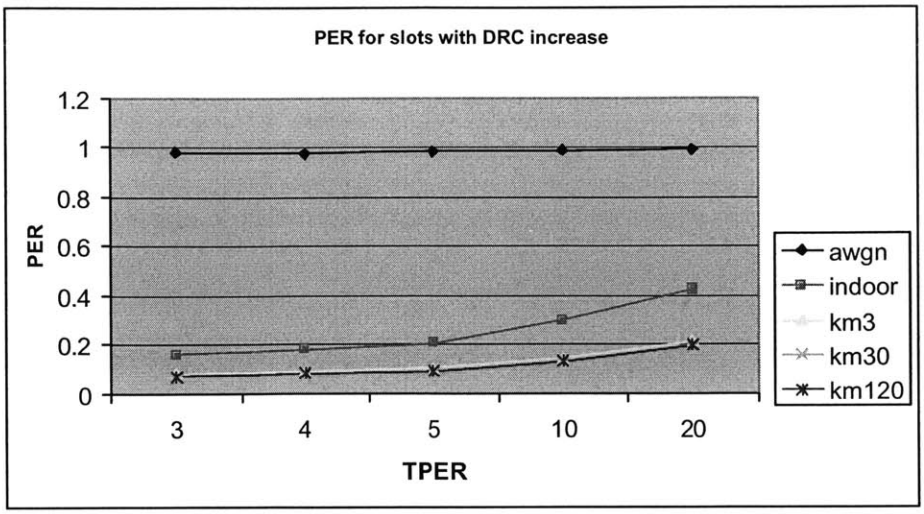


Figure 4.2-9: PER for the slots for which the DRC increased

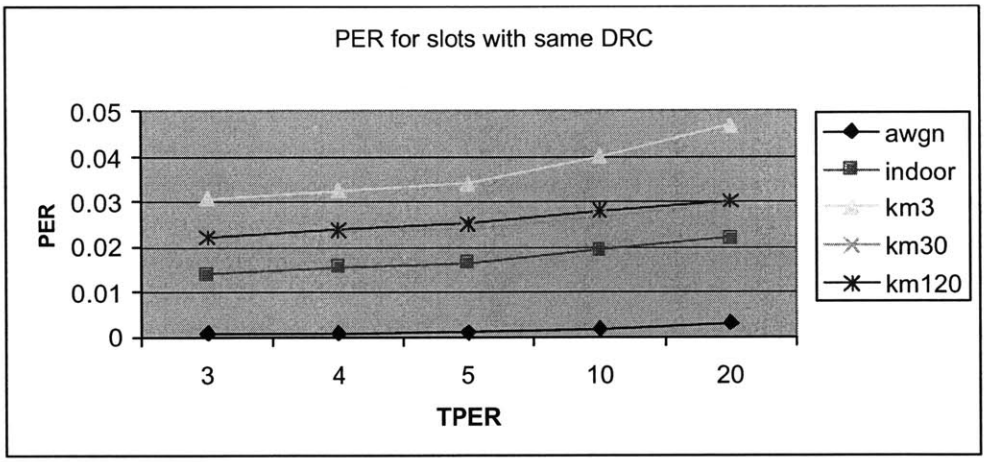


Figure 4.2-10: PER for the slots for which the DRC stayed the same

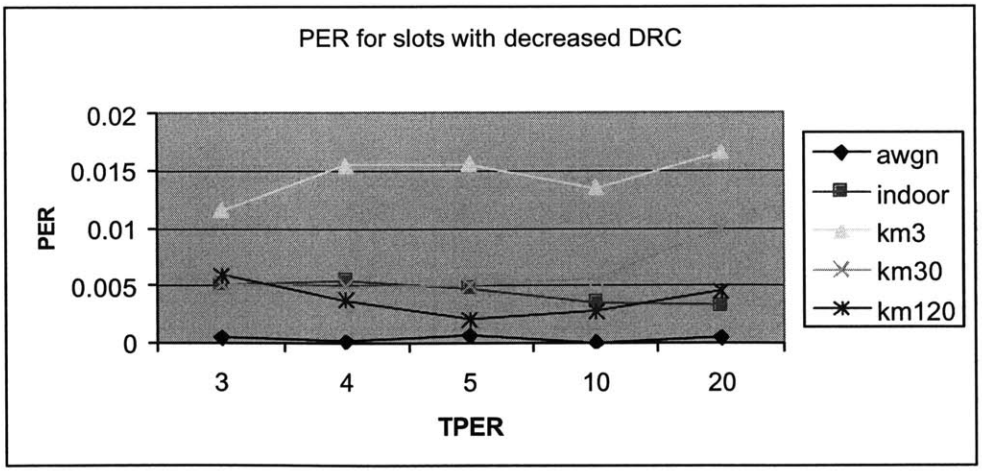


Figure 4.2-11: PER for the slots for which the DRC decreased

The physical layer simulations use a hard threshold when deciding on physical layer packet errors, since the PER-SINR characteristics of the codes used are very sensitive, as shown in Figure 3.2-1. With less steep curves, and soft decisions (as in real implementations), the throughput improvements would be greater. We already know how the average physical layer throughput and average physical layer packet error rate change as the TPER changes. However, it is unlikely that just two numbers can characterize the physical layer. In section 7.2, there is a more comprehensive analysis of the physical layer data rates and distributions.

4.2.2 TCP throughput with PPER and RLP Persistency

As the target error rate increases, the inter-arrival times of the packet errors decreases, but the lengths of error bursts do not change much. From the analysis it seems like the errors do not come in bursts. This also suggests that using a Bernoulli distribution for modeling the physical layer errors is reasonable. In addition, as the number of users increase, the scheduling separates the packet transmissions of a user, making the errors even less correlated. Let's look at how the TCP throughput depends on PPER and maximum allowed RLP retransmissions at different target packet error rates, for the km30 channel model. The physical layer packet error rate in these simulations represents the probability that a given packet will be unsuccessful. Also, the ratio of the number of lost packets to the total number of packets transmitted should approach the PER in the long run. These simulations model the errors as a Bernoulli process.

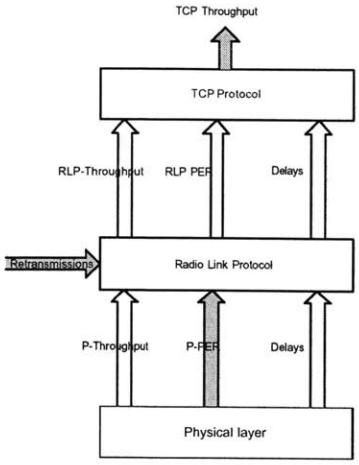


Figure 4.2-12: Throughput versus Physical layer Packet Error Rate and RLP Persistency

Figure 4.2-13 shows the TCP throughput for the km30 channel model with different target packet error rates.[24] These figures are similar to each other showing that the average PPER and max_rlp_retr are the main parameters of the TCP throughput. However they are not exactly the same, indicating that this simple model does not capture the physical layer characteristics. Figure 4.2-14 shows the throughput estimated by the model of this section, as in Figure 4.2-1. We see that again this estimation is very similar to the simulated throughput of the system. However, there are differences, showing that this model is only an approximation and that there are secondary effects not take into account.

[24] These OPNET simulations were varying the PER modeled as a Bernoulli process. Also the early terminations possible in the real system were not part of any of the simulations in the earlier sections.

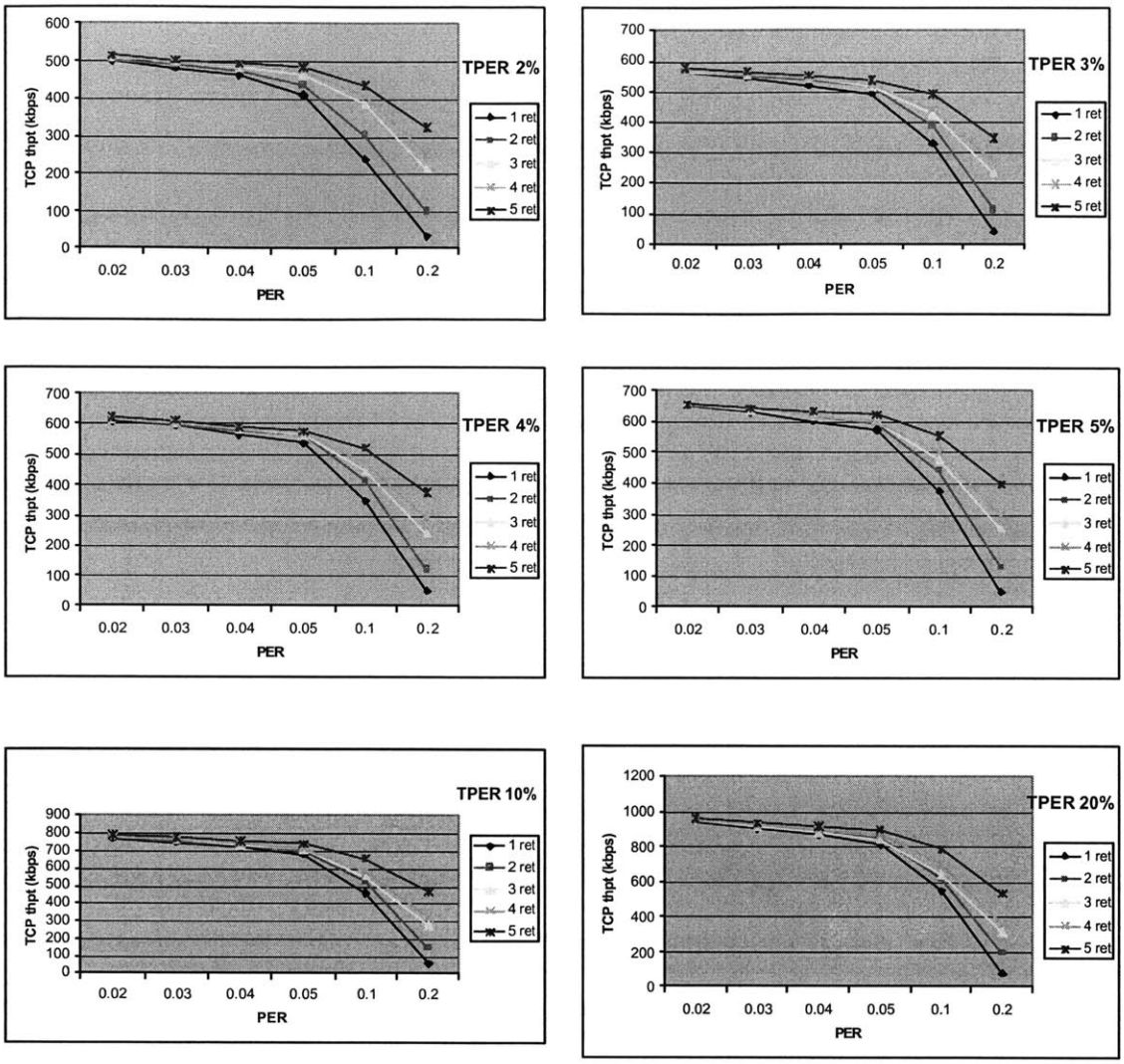
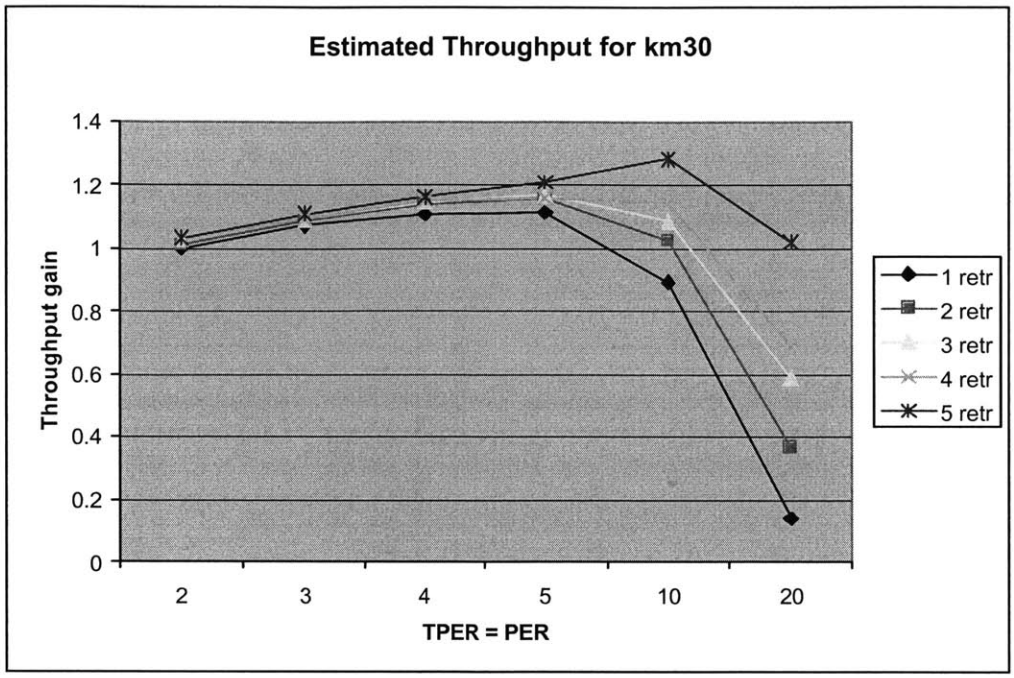


Figure 4.2-13: TCP throughput for the km30 channel



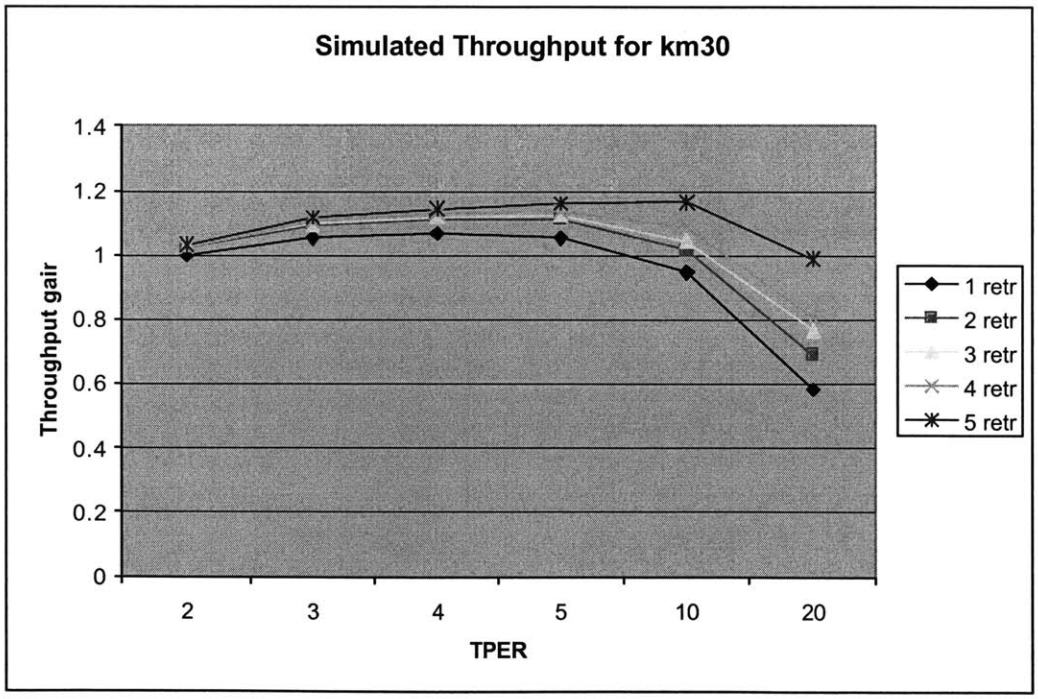


Figure 4.2-14: Estimated throughput gain in km30 channel

Chapter 5: Conclusion and Future Work

In this thesis, the forward link of a CDMA High Data Rate (HDR) system was analyzed using OPNET and physical layer simulation tools. The trade-off between error recovery at the physical layer and the radio link protocol (RLP) was studied for various channel conditions and a target packet error rate for optimal system performance was found. In addition, the HDR system layers were characterized. The issues of interest were the throughput, packet error rate, and delay. A simplified model with a few well-defined interfaces between these layers was used to explain system behavior.

A target packet error rate of 0.02 and single RLP retransmissions is an optimal combination for overall HDR performance. Increasing the target packet error rate increases both the physical layer throughput and the physical layer packet error rate. However, above a target packet error rate of 0.02, the throughput gain at the physical layer does not compensate for the throughput loss due to the TCP layer. Decreasing the target packet error rate lowers the physical layer throughput. This throughput decrease is not compensated for by the throughput increase due to the reduced packet error rate at the TCP layer, since 0.02 PER is low enough for efficient TCP performance.

Empirical models were developed from the characterizations of the system layers. The adjustable parameters of the models were the target packet error rate at the physical layer and the number of retransmissions at the RLP layer. Interactions between the physical layer and RLP-TCP layer can be modeled through the average throughput and average packet error rate. These models when combined explain system behavior to a certain degree. The discrepancy can be due to the delay characteristics, and non-uniform packet error rate at the physical layer.

For a more accurate model, further work needs to be done by including additional characteristics of the physical layer such as error durations and inter-arrivals. By analyzing the mechanisms in the system, we can develop a complete analytical model. We already did some work on this and modeled some parts of the system. It would be an interesting problem to mathematically predict the throughput of the system, given a number of users, channel conditions, and system parameters. Also when analyzing the sensitivity of RLP-TCP performance to the physical layer error rate the error rates were kept constant. Future work needs to be done on the affect of short durations of high error rate periods on the system throughput, since this behavior is often observed in real systems.

Bibliography

Digital Communications

[1] Sklar, Bernard. *Digital Communications: Fundamentals and Applications.* Englewood Cliffs, NJ: Prentice Hall. 1988.

[2] Proakis, John G. and Salehi, Masoud. *Communication Systems Engineering.* Upper Saddle River, NJ: Prentice Hall. 1994.

[3] Proakis, John G. *Digital Communications.* McGraw-Hill. 3rd Edition, 1995.

[4] Gibson, Jerry D. (editor). *The Mobile Communications Handbook.* IEEE Press. 1996

Internet RFCs related to TCP/IP

[5] User Datagram Protocol. J. Postel. Aug-28-1980.

[6] Internet Protocol. J. Postel. Sep-01-1981.

[7] Internet Control Message Protocol. J. Postel. Sep-01-1981. Transmission Control Protocol. J. Postel. Sep-01-1981.

[8] RFC 0872: TCP-on-a-LAN. M.A. Padlipsky. Sep-01-1982.

[9] RFC 1011: Official Internet protocols. J.K. Reynolds, J. Postel. May-01-1987.

[10] RFC 1072: TCP extensions for long-delay paths. V. Jacobson, R.T. Braden. Oct-01-1988.

[11] RFC 1122: Requirements for Internet hosts - communication layers. R.T. Braden. Oct-01-1989.

[12] RFC 1123: Requirements for Internet hosts - application and support. R.T. Braden. Oct-01-1989.

[13] RFC 1180: TCP/IP tutorial. T.J. Socolofsky, C.J. Kale. Jan-01-1991.

[14] RFC 1185: TCP Extension for High-Speed Paths. V. Jacobson, R.T. Braden, L. Zhang. Oct-01-1990.

[15] RFC 1323: TCP Extensions for High Performance. V. Jacobson, R. Braden, D. Borman. May 1992.

[16] RFC 1661: The Point-to-Point Protocol (PPP). W. Simpson, Editor. July 1994.

[17] RFC 1812 Requirements for IP Version 4 Routers. F. Baker. June 1995

[18] RFC 1812 Requirements for IP Version 4 Routers. F. Baker. June 1995.

[19] RFC 2001 TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms. W. Stevens. January 1997.

[20] RFC 2018 TCP Selective Acknowledgement Options. M. Mathis, J. Mahdavi, S. Floyd, A. Romanow. October 1996.

[21] RFC 2119 Key words for use in RFCs to Indicate Requirement Levels. S. Bradner. March 1997.

[22] RFC 2147 TCP and UDP over IPv6 Jumbograms. D. Borman. May 1997.

[23] RFC 2414 Increasing TCP's Initial Window. M. Allman, S. Floyd, C. Partridge. September 1998.

[24] RFC 1812 Requirements for IP Version 4 Routers. F. Baker. June 1995.

[25] RFC 2001 TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms. W. Stevens. January 1997.

[26] RFC 2018 TCP Selective Acknowledgement Options. M. Mathis, J. Mahdavi, S. Floyd, A. Romanow. October 1996.

[27] RFC 2119 Key words for use in RFCs to Indicate Requirement Levels. S. Bradner. March 1997.
[28] RFC 2147 TCP and UDP over IPv6 Jumbograms. D. Borman. May 1997.
[29] RFC 2414 Increasing TCP's Initial Window. M. Allman, S. Floyd, C. Partridge. September 1998.
[30] RFC 2581 TCP Congestion Control. M. Allman, V. Paxson, W. Stevens. April 1999.

TCP Related Papers

[31] Ludwig, Eric Reiner, Eliminating Inefficient cross-layer interactions in wireless networking, Dissertation, 04.04.2000
[32] Paxson, Vern, Measurement and analysis of End-to-End Internet Dynamics, PhD Thesis, University of California Berkeley, April 1997.
[33] Shepard, Timothy Jason. TCP Packet Trace Analysis, Master of Science thesis, MIT, USA, June 1990
[34] Balakrishnan. Hari, Padmanabhan, Venkata N. IETF internet draft, TCP Performance Implications of Network Asymmetry, March 2000
[35] H. Balakrishnan, "Challenges to Reliable Data Transport over Heterogeneous Wireless Networks", Ph.D. Thesis, University of California at Berkeley, USA, August 1998 http://www.cs.berkeley.edu/~hari/thesis/
[36] H. Balakrishnan, "Explicit Loss Notification and Wireless Web Performance"
[37] Berners-Lee Tim, Cailliau Robert, Nielsen Henrik Frystyk, Secret Arthur, The World-wide web, Communications of the ACM, pages 76-82, August 1994/Vol 37 No8
[38] Floyd Sally, TCP and Successive Fast Retransmissions, May 1995
[39] Jacobson Van, end2end-interest mail: modified TCP congestion avoidance algorithm, 30 April 1990
[40] Jacobson, V. Congestion avoidance and control. In Proceedings of SIGCOMM '88 (Stanford, CA, Aug 1988), ACM.
[41] Karn, P. and C. Partridge, "Estimating Round-Trip Times in Reliable Transport Protocols", Proc. SIGCOMM '87, Stowe, VT, August 1987.
[42] Padmanabhan, Venkata N. Caceres, Ramon. Fast and Scalable Handoffs for Wireless Internetworks, MOBICOMM 96, Rye NY USA
[43] Wu, Gang. Bai, Yong. Lai, jie and Ogielski, Andy. Interactions between TCP and RLP in Wireless Internet, Globecomm'99
[44] Bao, Gang. Performance evaluation of TCP/RLP protocol stack over CDMA wireless link, Wireless Networks, pp229-237, 1996
[45] Zhang, L., "Why TCP Timers Don't Work Well", Proc. SIGCOMM '86, Stowe, Vt., August 1986.
[46] Paxson, Vern, End-to-End Internet Packet Dynamics, SIGCOMM '97. June 23, 1997.
[47] Mogul, Jeffrey C, Observing TCP Dynamics in Real Networks, WRL research report 92/2. April 1992.
[48] Kay Jonathan and Pasquale Joseph, The importance of non-data touching processing overheads in TCP/IP, SIGCOMM '98.
[49] Chockalingam,A.; GangBao, Performance of TCP/RLP protocol stack on correlated fading DS-CDMA wireless links, Vehicular Technology, IEEE Transactions on , Volume: 49 Issue: 1 , Jan.2000 Page(s): 28 –33

HDR Related Papers

[50] Paul Bender et al. "CDMA/HDR: A bandwidth efficient high-speed data service for nomadic users," IEEE Communications Magazine, Vol.38, pp70-77, July 2000

[51] A. Jalali et al "Data throughput of CDMA/HDR a high efficiency-high data rate personal communication wireless system," in Proc. IEEE 51st Vehicular Technology Conference, Tokyo, Japan, May 2000
[52] The HDR standard: Cdma2000 – 1xEV-DO Air interface Specification, to appear. http://www.3gpp2.org
[53] Esteves, Eduardo. The High Data Rate Evolution of the cdma2000 Cellular System, to appear in MMT2000 Conference in December
[54] Cai, Jian. Goodman, David J. General Packet Radio Service in GSM, In IEEE Communications Magazine, October 1997, pages 122-131

Tools used in simulation, data analysis

[55] Pradeep K. Singh. White paper: TCP Modeling in OPNET, March 4, 2000
[56] OPNET user guide, OPNET reference books
[57] Shawn Ostermann. *tcptrace -TCP dump file analysis tool.*, http://jarok.cs.ohiou.edu/software/tcptrace/tcptrace.html
[58] Chariot user guide

Reference Books

[59] Comer, Douglas E. Internetworking with TCP/IP Vol I: Principles, Protocols, and Architecture, fourth edition, Prentice Hall, New Jersey, 2000

Appendix

7.1 Channel Models

The parameters used in the channel simulations, are in Table 10, for the interested readers.

Table 10: Channel Simulation Parameters

Parameter	AWGN	Indoor	Km3	Km30	Km120
Num of antennas	1	1	1	1	1
Num of paths	1	2	1	1	1
Combiner	Optimal	Optimal	Optimal	Optimal	Optimal
Shadow Stan	8	8	8	8	8
Shadow Corr.	.5	.5	.5	.5	.5
Shadow on	1	1	1	1	1
Reyleigh on	0^M	1^M	0^M	0^M	0^M
Max allowable path loss	147	147	147	147	147
Eirp	350	350	350	350	350
Mobile antenna height	1.5	1.5	1.5	1.5	1.5
Base antenna height	20^M	20^M	20^M	20^M	20^M
Frequency reuse	13	16	13	13	13
Antenna correlation	0.5	0.5	0.5	0.5	0.5
Ricean K^M	100^M	-100^M	-100^M	-100^M	-100^M
Path gains	1.0	.75, .25	1.0	1.0	1.0
Distance over lambda	1	1	1	1	1
Number of antenna points	360	360	360	360	360
Lambda	0.15	0.15	0.15	0.15	0.15
Cell loading	1.0	1.0	1.0	1.0	1.0
Mobility indicator	1.	0	1	1	1
Num sector antennas	3	3	3	3	3
Sector to sector correlation	0	0	0	0	0
Antenna type	65	33	65	65	65
Ts_sec	0.000835	0.000835	0.000835	0.000835	0.000835
Building penetration loss	0	0	0	0	0
SINR limit	13.0	13.0	13.0	13.0	13.0

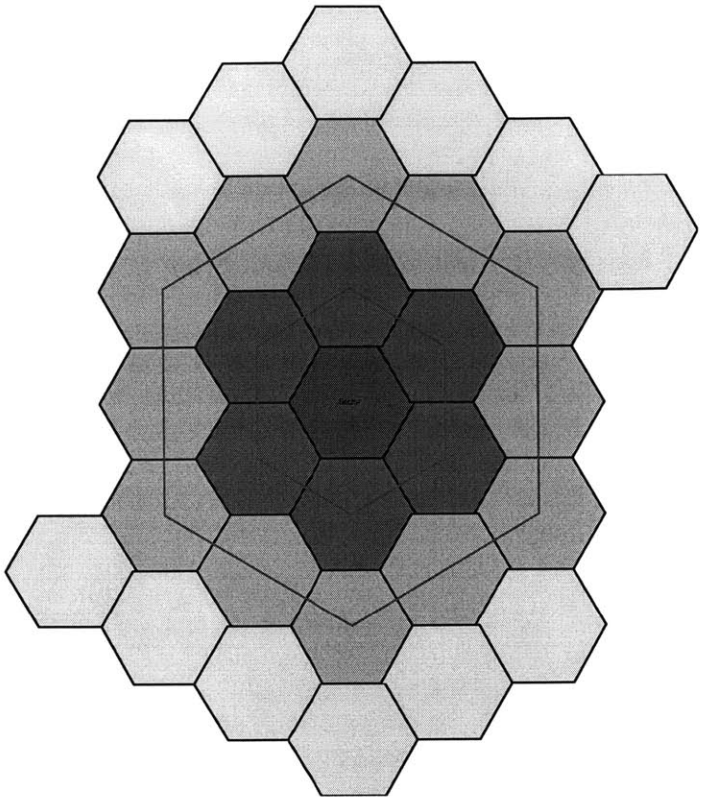
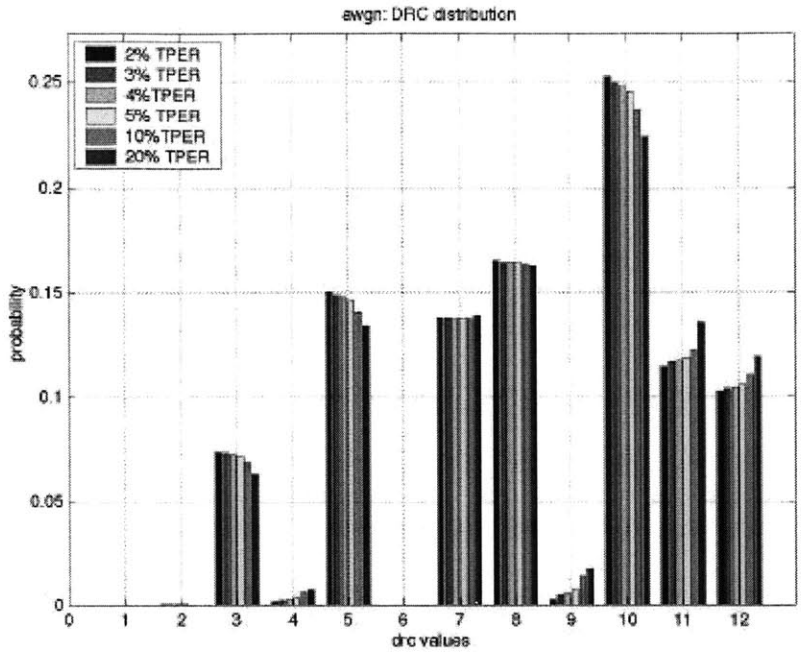
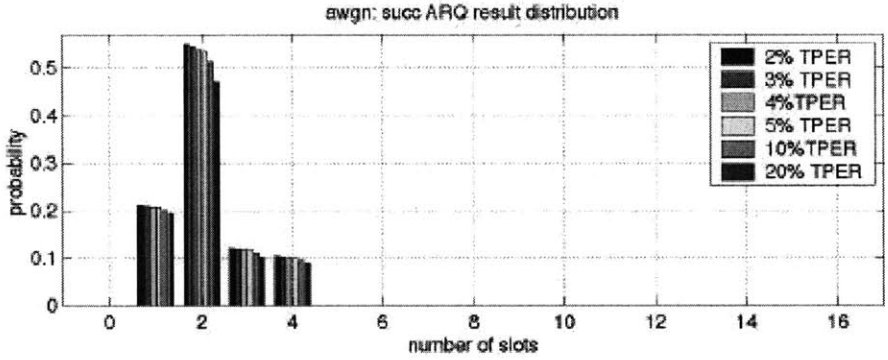


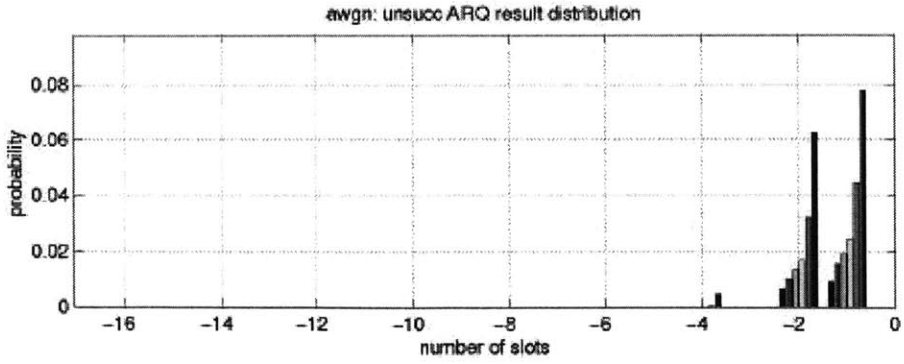
Figure 7.1-1: Embedded sector setup used in channel simulations

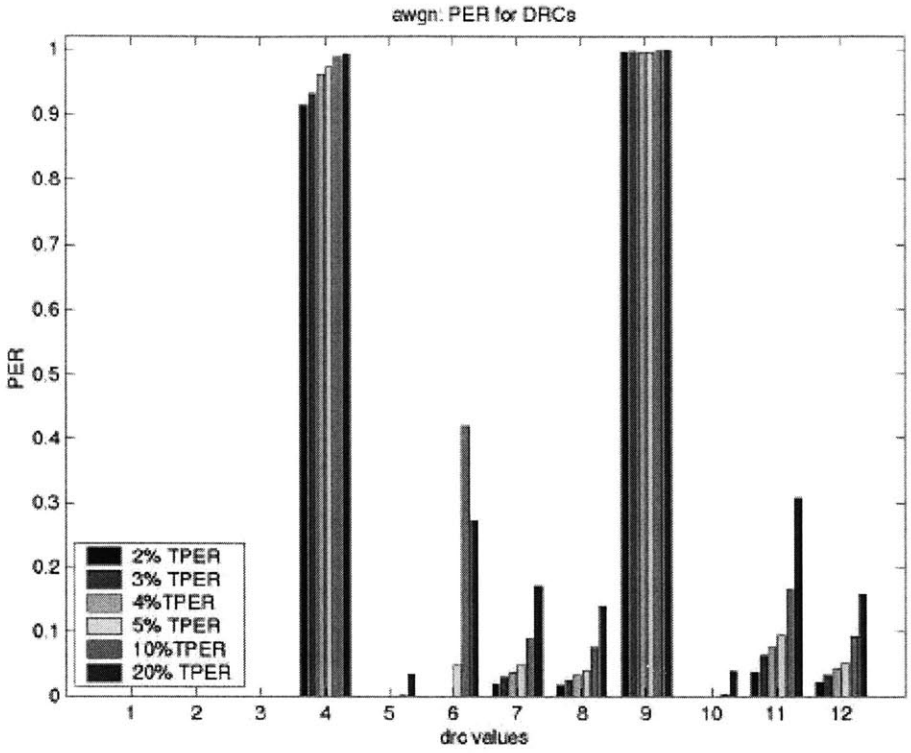
7.2 DRC distributions for different channel conditions

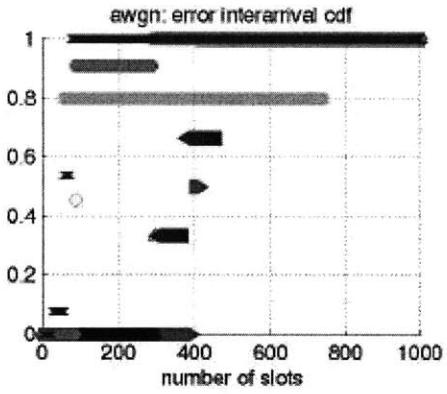
In this section detailed analysis is done on the output of the physical layer simulator. The output file includes the data rate requested by each user, the outcome of the physical layer packet if it was transmitted at that slot. The outcome can be read off as being just successful, or unsuccessful, as in the earlier sections of Chapter 4. Or the number of slots it took for the packet to be decoded successfully, in case of an early termination, can be read. There are four plots summarizing the important characteristics of the data generated for each channel condition with different TPER. The first of these show the distribution of the DRC values as in Table 3, and is useful in understanding how the data rates change as TPER changes. The second plots, show the ARQ outcome in a compact way. The positive numbers correspond to the number of slots required for successful transmissions and can vary from 1 to 16 due to early terminations. The negative numbers correspond to the number of slots used for unsuccessful transmissions and can only have values of 1,2, 4, 8, 16 since an unsuccessful transmission needs to go all the way. The third plot shows the packet error rates corresponding to different data rates, and the last plot shows the durations and inter-arrivals of errors and successes in units of number of slots.

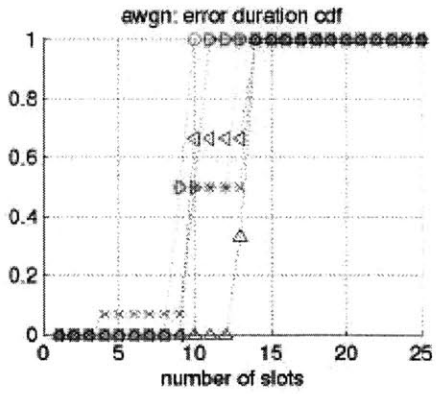

awgn: DRC distribution
2% TPER
3% TPER
4%TPER
5% TPER
10%TPER
20% TPER
probability
drc values

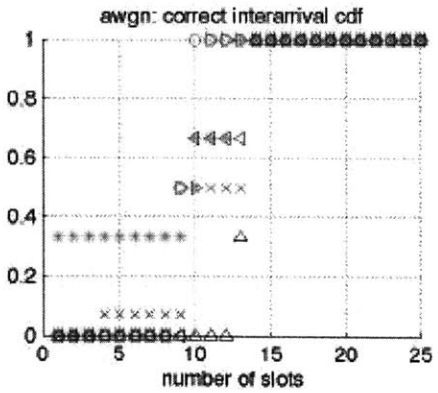

awgn: succ ARQ result distribution
2% TPER
3% TPER
4%TPER
5% TPER
10%TPER
20% TPER
probability
number of slots

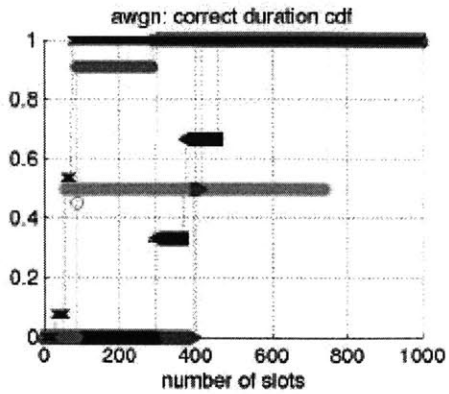

awgn: unsucc ARQ result distribution
probability
number of slots

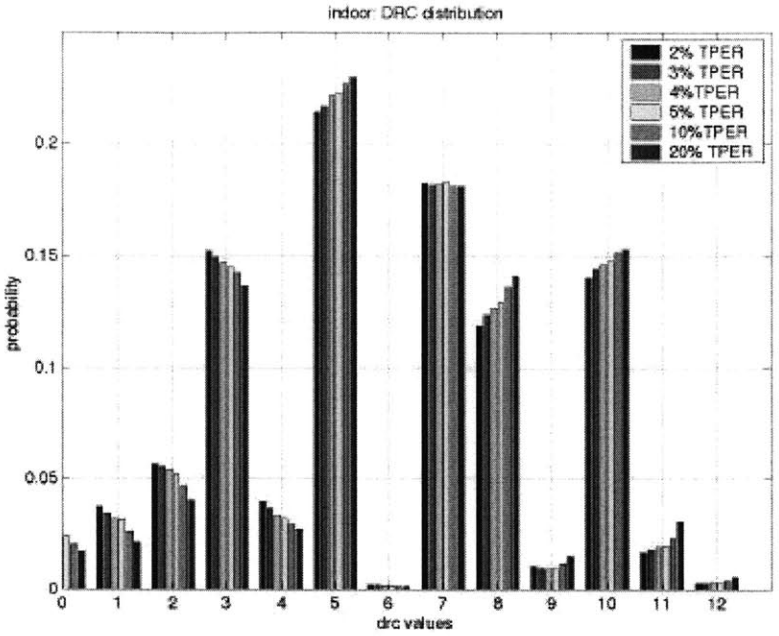

awgn: PER for DRCs
PER
drc values
2% TPER
3% TPER
4%TPER
5% TPER
10%TPER
20% TPER

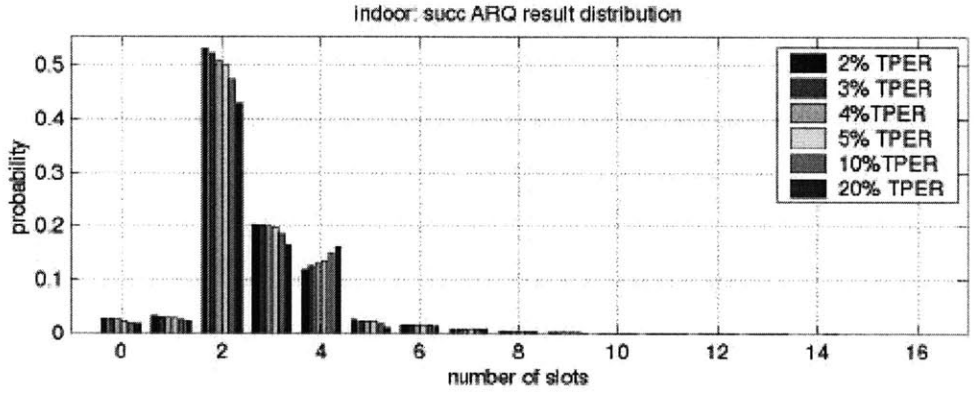

awgn: error interarrival cdf
number of slots

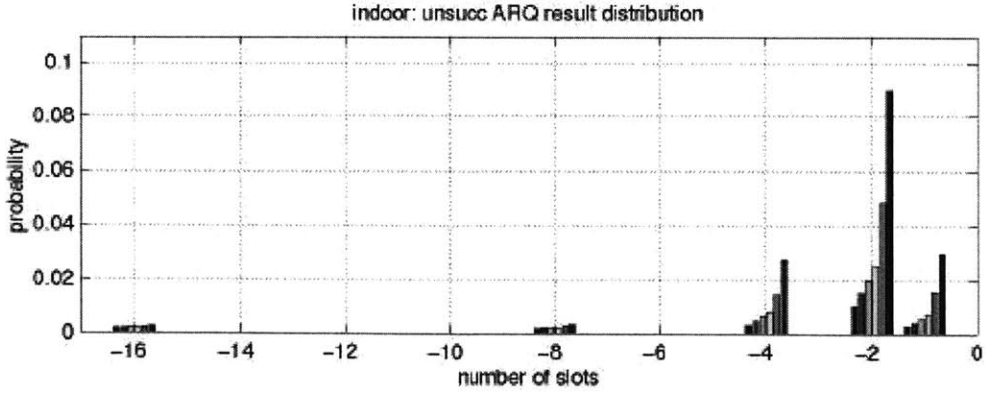

awgn: error duration cdf
number of slots

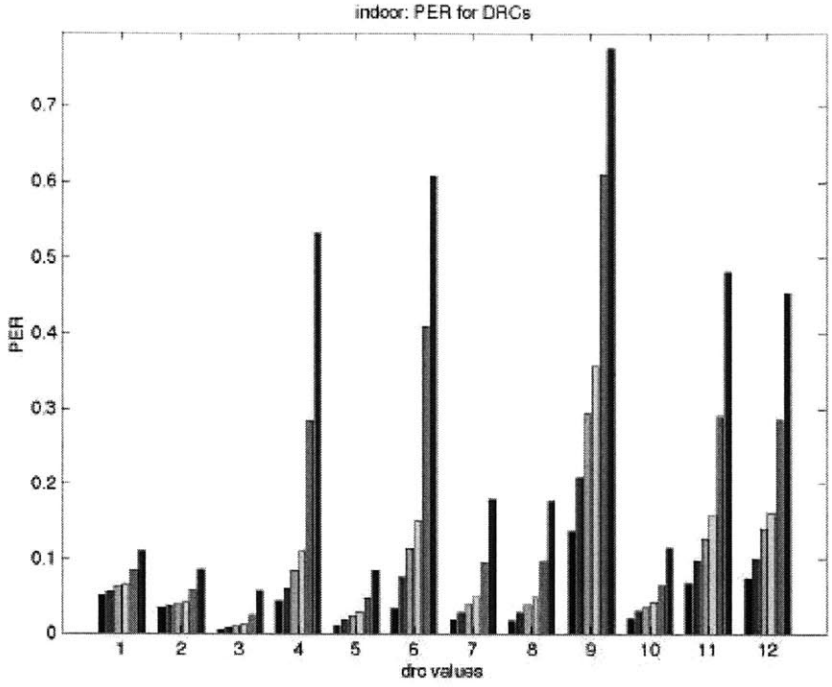

awgn: correct interarrival cdf
number of slots

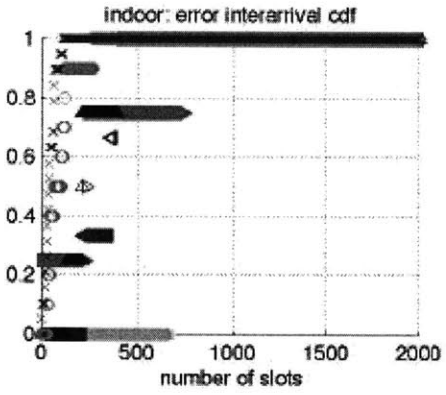

awgn: correct duration cdf
number of slots

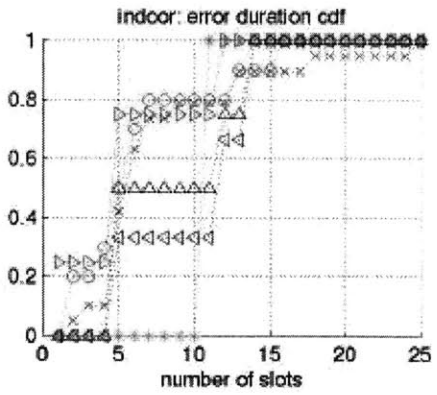

indoor: DRC distribution
2% TPER
3% TPER
4%TPER
5% TPER
10%TPER
20% TPER
probability
drc values

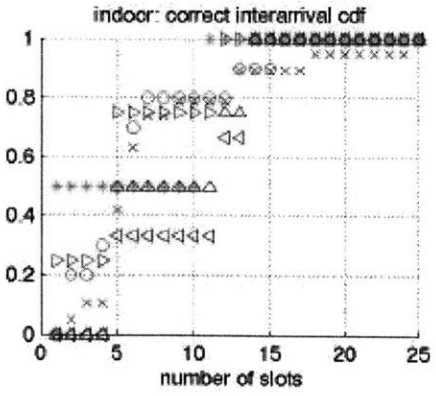

indoor: succ ARQ result distribution
2% TPER
3% TPER
4%TPER
5% TPER
10%TPER
20% TPER
probability
number of slots

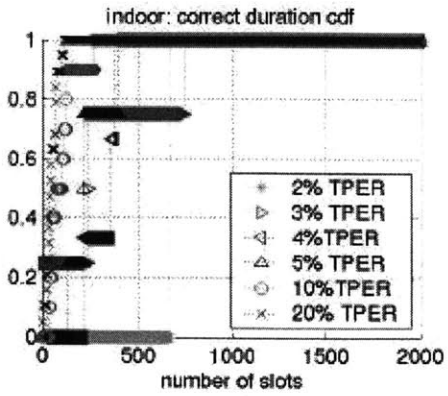

indoor: unsucc ARQ result distribution
probability
number of slots

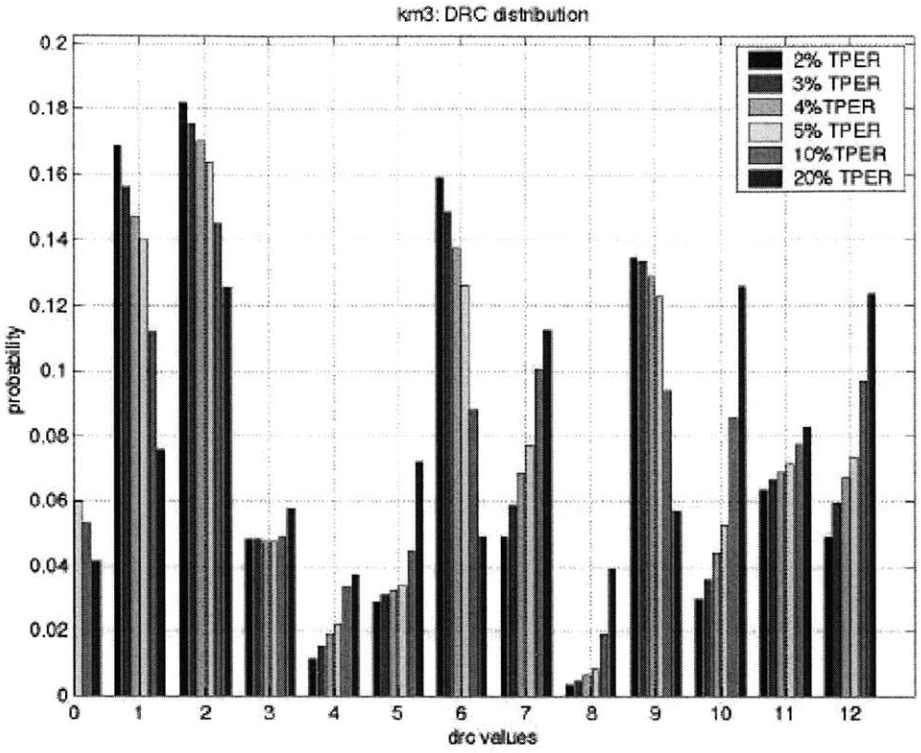

indoor: PER for DRCs
PER
drc values

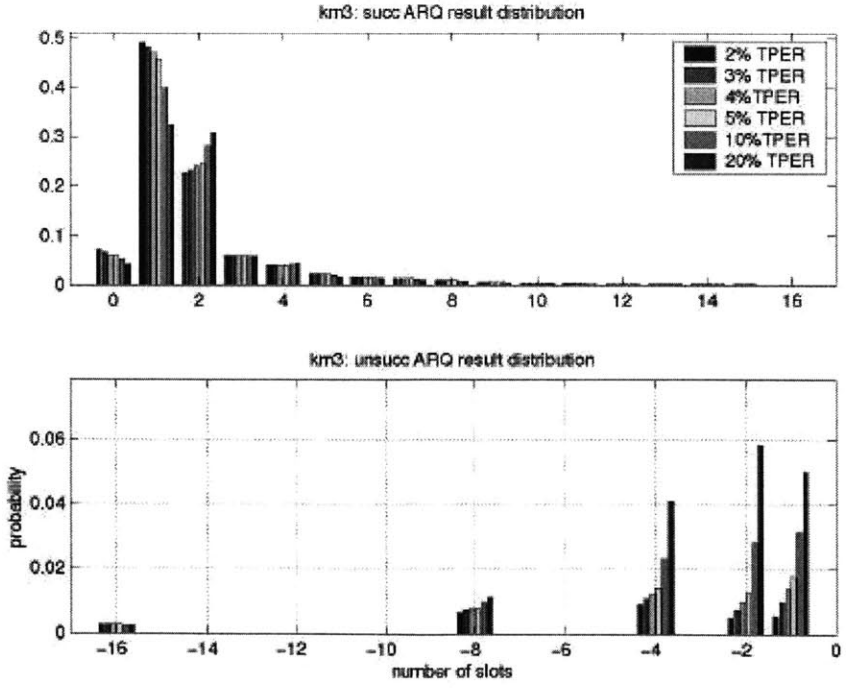

indoor: error interarrival cdf
number of slots

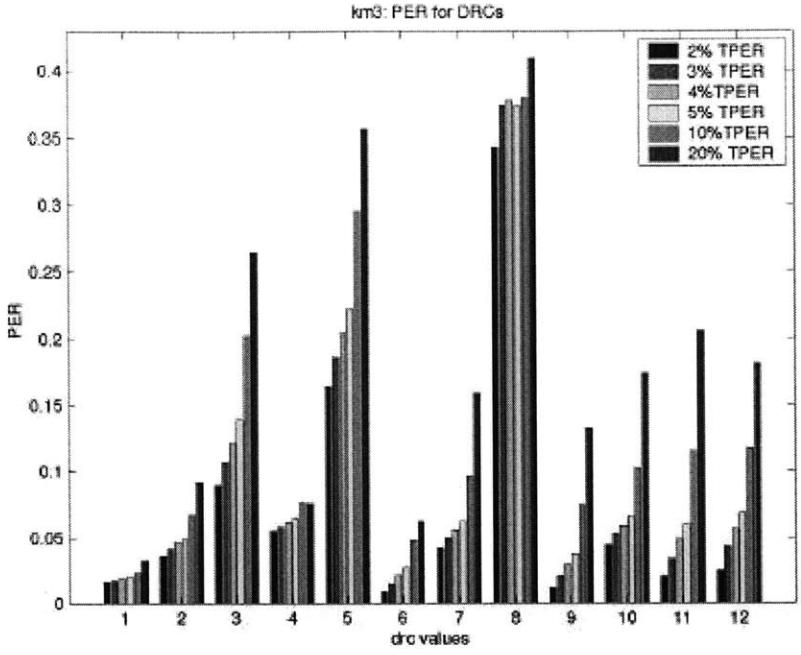

indoor: error duration cdf
number of slots

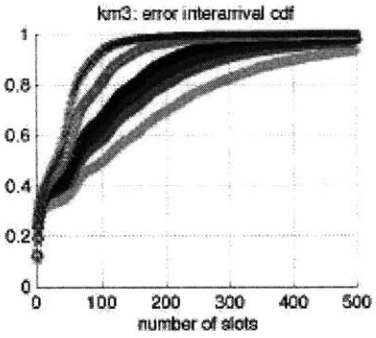

indoor: correct interarrival cdf
number of slots

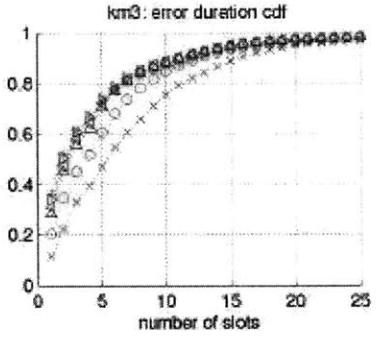

indoor: correct duration cdf
2% TPER
3% TPER
4%TPER
5% TPER
10%TPER
20% TPER
number of slots

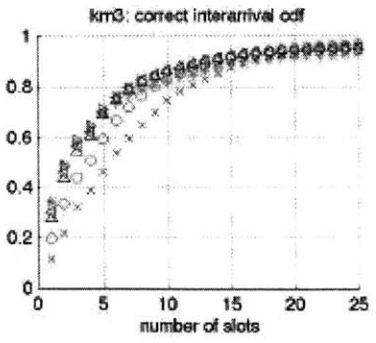

km3: DRC distribution
2% TPER
3% TPER
4%TPER
5% TPER
10%TPER
20% TPER
probability
drc values

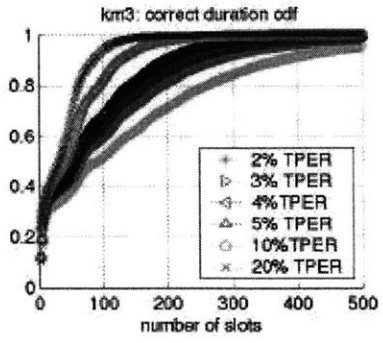

km3: succ ARQ result distribution
2% TPER
3% TPER
4%TPER
5% TPER
10%TPER
20% TPER
km3: unsucc ARQ result distribution
probability
number of slots

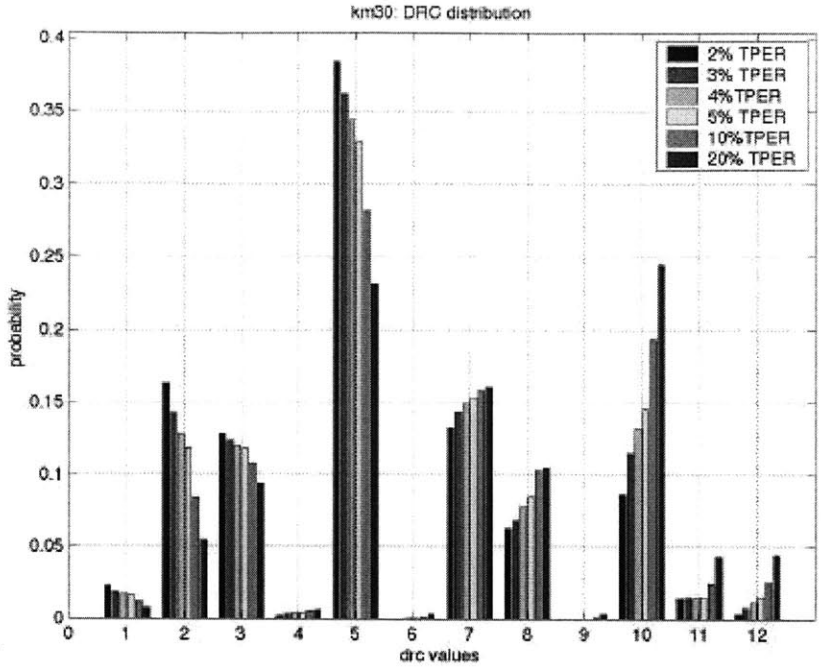

km3: PER for DRCs
2% TPER
3% TPER
4%TPER
5% TPER
10%TPER
20% TPER
PER
drc values

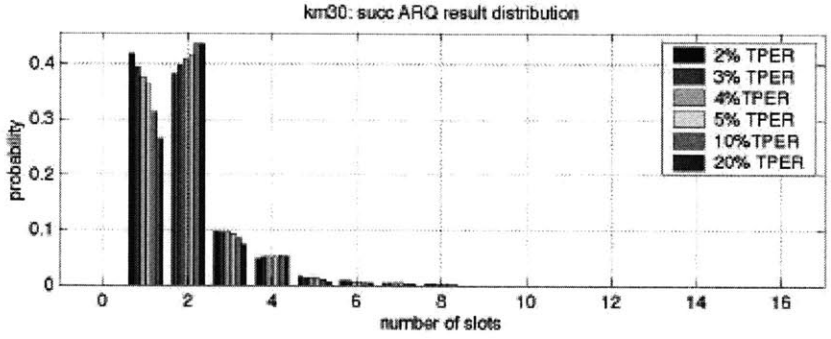

km3: error interarrival cdf
number of slots

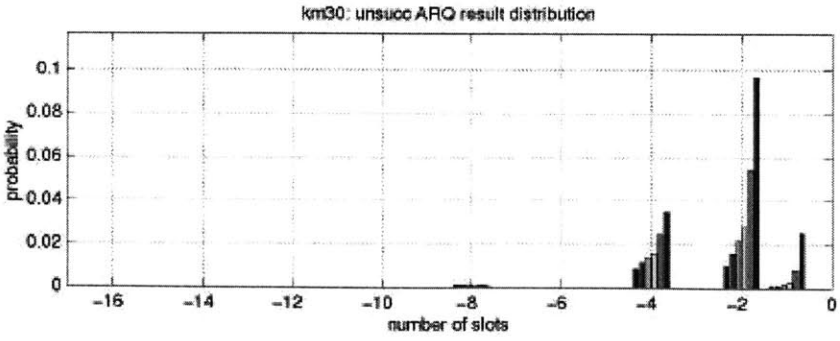

km3: error duration cdf
number of slots

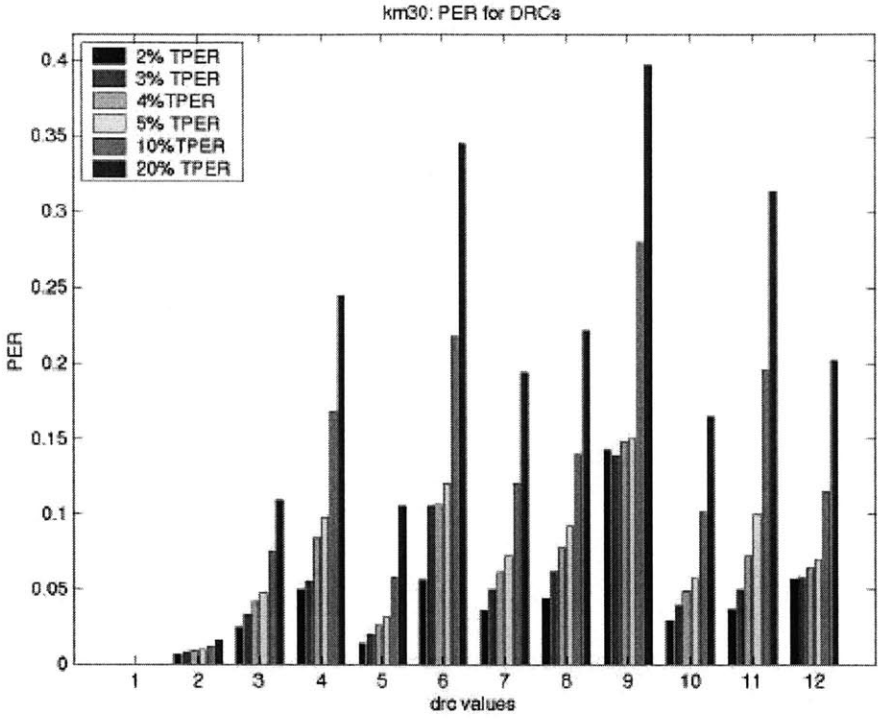

km3: correct interarrival cdf
number of slots

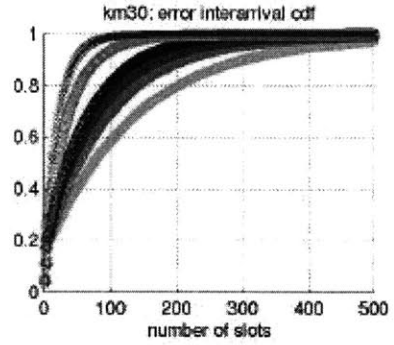

km3: correct duration cdf
2% TPER
3% TPER
4%TPER
5% TPER
10%TPER
20% TPER
number of slots

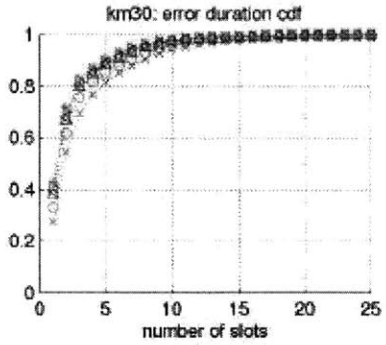

km30: DRC distribution
2% TPER
3% TPER
4%TPER
5% TPER
10%TPER
20% TPER
probability
drc values

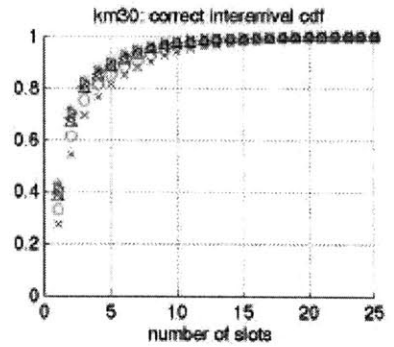

km30: succ ARQ result distribution
2% TPER
3% TPER
4%TPER
5% TPER
10%TPER
20% TPER
probability
number of slots

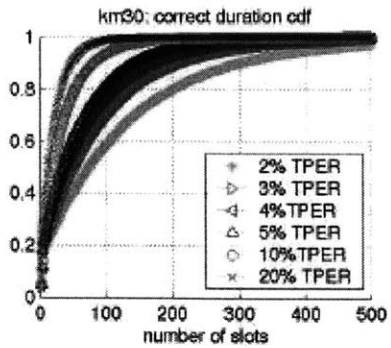

km30: unsucc ARQ result distribution
probability
number of slots

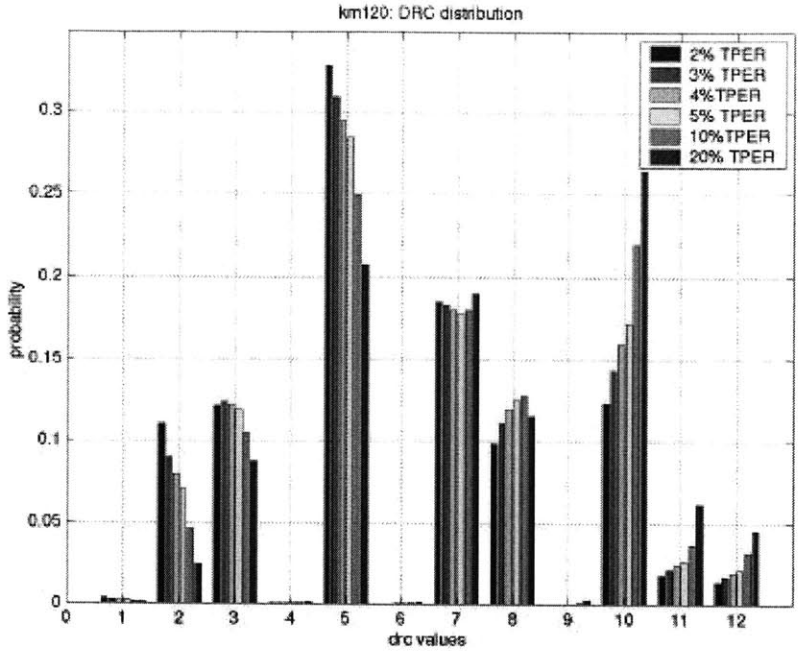

km30: PER for DRCs
2% TPER
3% TPER
4%TPER
5% TPER
10%TPER
20% TPER
PER
drc values

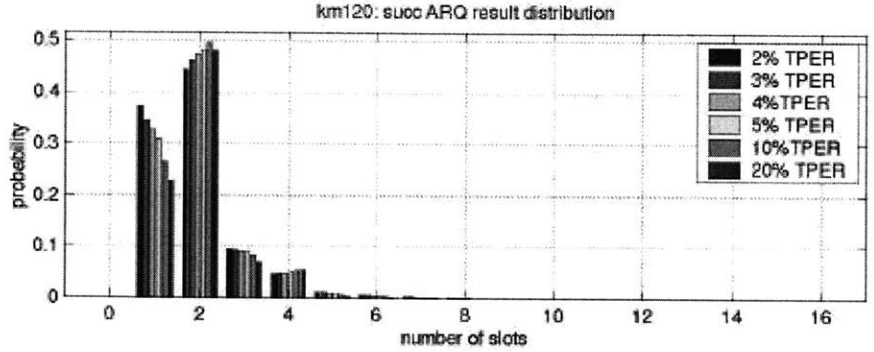

km30: error interarrival cdf
number of slots

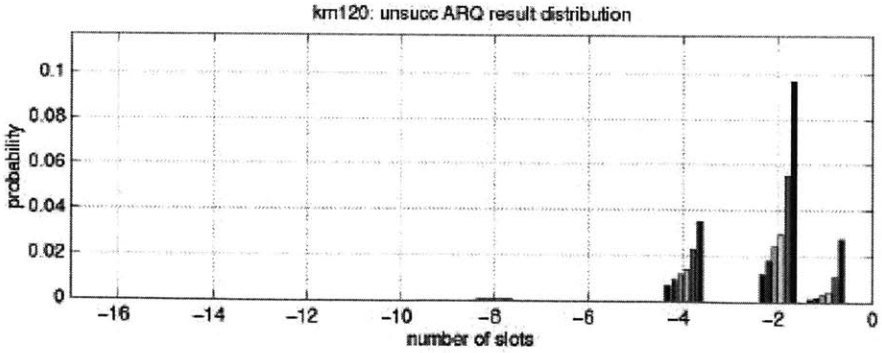

km30: error duration cdf
number of slots

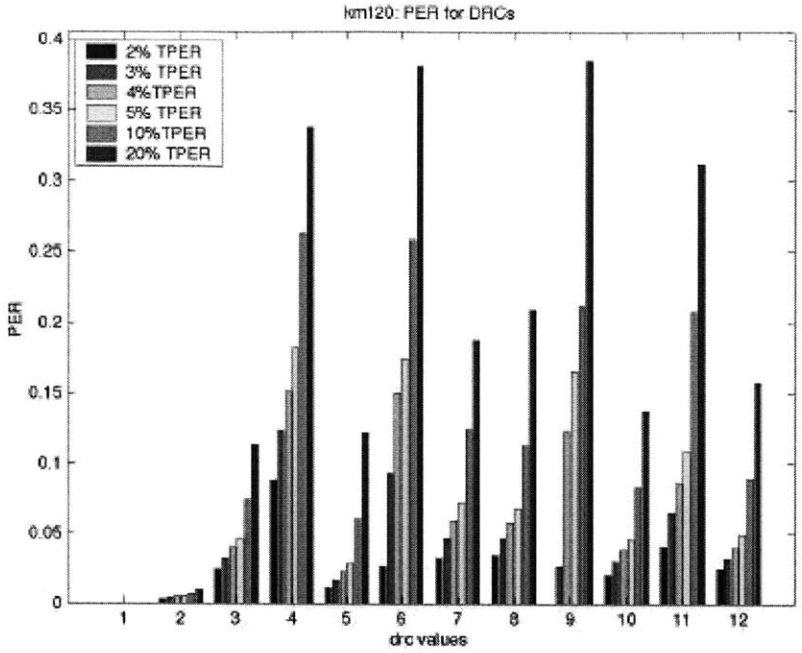

km30: correct interarrival cdf
number of slots

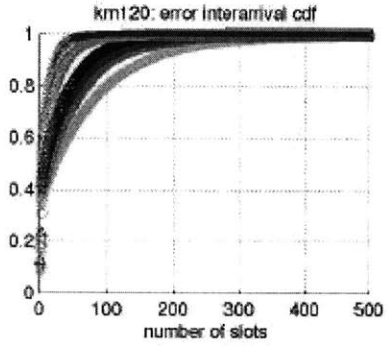

km30: correct duration cdf
2% TPER
3% TPER
4%TPER
5% TPER
10%TPER
20% TPER
number of slots

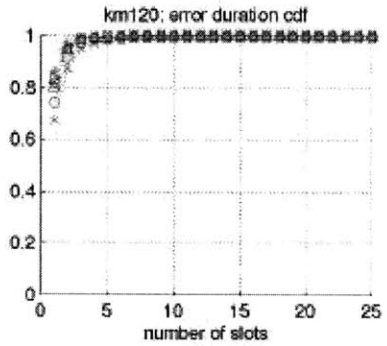

km120: DRC distribution
2% TPER
3% TPER
4%TPER
5% TPER
10%TPER
20% TPER
probability
drc values

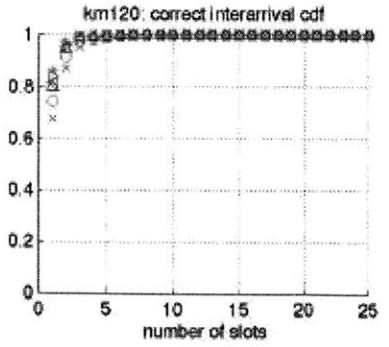

km120: succ ARQ result distribution
2% TPER
3% TPER
4%TPER
5% TPER
10%TPER
20% TPER
probability
number of slots

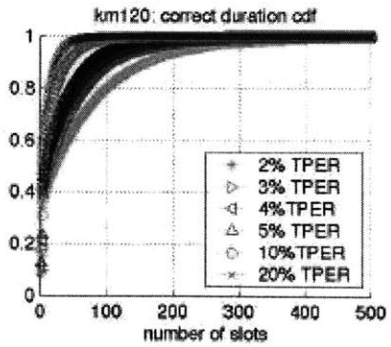

km120: unsucc ARQ result distribution
probability
number of slots


km120: PER for DRCs
2% TPER
3% TPER
4%TPER
5% TPER
10%TPER
20% TPER
PER
drc values


km120: error interarrival cdf
number of slots


km120: error duration cdf
number of slots


km120: correct interarrival cdf
number of slots


km120: correct duration cdf
2% TPER
3% TPER
4%TPER
5% TPER
10%TPER
20% TPER
number of slots