UPPSALA UNIVERSITY

On Scheduling and Adaptive Modulation in Wireless Communications

Nilo Casimiro Ericsson

June 2001

SIGNALS AND SYSTEMS
UPPSALA UNIVERSITY
UPPSALA, SWEDEN

Submitted to the Faculty of Science and Technology, Uppsala University in partial fulfillment of the requirements for the degree of Technical Licentiate in Signal Processing.

Printed in Sweden by Lindbergs Grafiska, Uppsala, 2001

To Britney

Abstract

This thesis addresses problems appearing in the transition from a fixed Internet to a mobile and wireless one. The focus is on scheduling of downlink transmissions of packet data over wireless links with varying quality, due to multipath fading.

Recent work suggests that predictions of channel quality can be achieved for several milliseconds ahead in time, for fast moving (vehicular) users. In this thesis we try to answer the question of how accurate, or, how long predictions we need in order to make efficient resource allocations through time-slot scheduling, and adaptive modulation.

An alternative approach to a deterministic scheduling of link resources, is the usage of incremental redundancy, or so-called Hybrid type-II ARQ. This thesis suggests an approach where predictive scheduling is combined with incremental redundancy, to obtain a fast, and robust, predictive resource allocation scheme.

The conclusions drawn are mainly: 1) A prediction of the signal-to-interference ratio with error standard deviation of up to 3.5 dB significantly improves the quality of the link. It makes it possible to maintain a nearly error free transmission at a low additional cost in delay, when using a simple time-slot scheduler with Hybrid type-II ARQ. 2) For error-sensitive applications, utilizing TCP/IP, a robust link layer protocol with ARQ should be used to avoid expensive transport layer re-transmissions.

These conclusions are drawn from network- and link layer simulations, and are based on expected performance at the transport layer.

Acknowledgments

First of all, thanks to my two main supervisors, Prof. *Anders Ahlén* and Prof. *Mikael Sternad*. You make this work interesting, mainly by indicating how close the people at Ericsson are to start working with the problems we are trying to solve: "They are close on our heels, and once they see the potential..." Anders and Mikael also provide an extremely creative and fun work environment, together with all the people in the Signals & Systems group at Magistern.

The ones I've been working most closely with at Magistern have my gratitude for being so great *colleagues*, especially the old-boys *Torbjörn Ekman*, *Jonas Öhr*, and *Tomas Olofsson*. Who knows, we might even become *friends* one day. `;-)` Keep restraining the seriousity level!

Your computer sucks, who you gonna call? Sysadmin *Ove Ewerlid* not only listens to your computer complaints, but also helps with whatever is bothering you, and often with some words of wisdom, attached for future reference.

This thesis would not have been possible to finalize without the persistent help from, and fruitful cooperation with, Tekn. Lic. *Sorour Falahati*, at Signals & Systems, Chalmers University of Technology. Thank you Sorour, for enduring long pre-deadline nights by the keyboard, running fat simulations, and including last minute updates to the source code.

The work is supported by the Swedish Foundation of Strategic Research, through the PCC (Personal Computing and Communication) research program. The aim of PCC is to work towards personal multimedia communication to all, at the same cost as fixed telephony today. Within PCC, the Wireless IP (WIP) project develops innovative approaches to increase spectrum efficiency and throughput for packet data over wireless links.

All the other people in the PCC program, students and supervisors: I'm glad I got to know you, it has been interesting and it will become even more.

What is work worth without play? To all my friends and family in Uppsala and Norrköping, a great hug for playing with me, and sharing the fruit.

In case you were expecting to find your name on this page, and did not –

there's a sequel, and it's better...

Contents

Chapter 1

Introduction

This thesis addresses problems appearing in the transition from a fixed Internet to a mobile and wireless one. The focus is on scheduling of downlink transmissions of packet data over wireless links with varying quality, due to multipath fading.

Adaptivity on all levels is the most important means for achieving high bandwidth efficiency in the wireless link, and also satisfying the required communication quality for the served applications. The essence of this statement is that our communication system must be able to assign the scarce wireless resources where they are needed, and where they are expected to provide the most benefit. It is important not to waste the bandwidth on things such as repetitions of already transmitted information, overprotective channel coding, and information that the wireless terminal is unable to present. The statement also implies that the applications should be able to adjust their requirements on the requested services from the communication system.

Recent work by Torbjörn Ekman and others [15, 16, 56], suggests that predictions of channel quality can be achieved for several milliseconds ahead in time. In this thesis we try to answer the question of how accurate, or, how long predictions we need in order to perform efficient resource allocations through scheduling.

An alternative approach to a deterministic scheduling of link resources, is the usage of incremental redundancy, or so-called Hybrid ARQ type-II, studied by Sorour Falahati and others [22, 24, 25, 26]. This thesis also suggests an approach where predictive scheduling is combined with incremental redundancy, obtaining a fast, and robust, predictive resource allocation scheme.

The conclusions drawn are mainly that 1) for error-sensitive applications, utilizing TCP/IP, a robust link layer protocol with ARQ should be used, and 2) for less error-sensitive applications, utilizing UDP/IP, a less robust link layer can provide acceptable services, under certain circumstances. These conclusions are drawn from network- and link layer simulations, and are based on expected performance at the transport layer. More precise implications and suggestions for improvements will be studied next, where simulations will also cover the experienced quality by the users in a future personal communication system.

The thesis is organized as follows: Chapter 2 gives an introduction to fading channels. Chapters 3 through 5 describe the different layers of the OSI model, along with considerations for achieving quality of service (QoS) also on networks including wireless links. In Chapter 6 the scheduling approach suggested for future wireless links is introduced. Simulations are carried out to evaluate the impact of scheduling on the performance over wireless links in Internet communications. The simulations and their results are given in Chapter 7. Finally, the thesis is concluded in Chapter 8, where forthcoming efforts and work is also described.

1.1 Words, Acronyms, Abbreviations, and the OSI Model

The OSI reference model for computer communications is briefly described in Appendix B, a description that can be read by those not familiar with the subject.

Many words and expressions circulate in the world of computer- and telecommunications. Appendix C is intended to give some guidance in this jungle. Acronyms are written in their full length in the Abbreviations and Acronyms section, C.2, and the meaning is described. Some of the included words are also explained in the Words section, C.1.

Chapter 2

Fading Channels

The performance of a wireless digital communication system is dictated by many parameters. The *power* of the received radio signal is important, but several other factors also matter. A signal can have a very low power, but if the signal is present for a longer time, then the *energy* assembled by the receiver can be big enough to detect the transmitted symbol. The signal can also have wide-band characteristics, meaning that the radio signal energy is spread over a wider frequency span than the base-band signal would require. The *noise* level is also important, since it introduces a disturbance into the detection system, a disturbance that in some cases corrupts the detector's decision, and in some cases makes it impossible to acquire the radio signal at all. In a system with many radio transmitters, such as a wireless LAN or a GSM-system, the level of radio *interference* is high and requires control, in order not to cause disturbances. Moreover, in a mobile radio system, other effects, such as Doppler frequency shift and varying multipath propagation fading, contribute to the deterioration of the performance.

The received signal energy in relation to the noise present at the receiver dictates the signal detection performance. The relation is called the Signal-to-Noise Ratio (SNR). If there also is a substantial level of unknown interference from other transmitters, then the interference is regarded as noise. The relevant ratio is then called the Signal-to-Noise-and-Interference Ratio (SNIR). Since these energy levels are relative to the noise level, they are usually measured in decibel (dB), and they can also be normalized with respect to the rate of the transmitted data, for example the SNR per bit, or the SNIR per symbol.

This chapter is intended to give an introduction to the concept of fading in radio channels, and the remedies that exist to counteract the problems it imposes on the communication system. First, a simple analogy is given to multipath fading, along with a real-life scenario. Then it is discussed whether the fading could be predicted within a reasonable scope of time. In Section 2.3 it is discussed what type of approach should be adopted in the battle against fading; averaging or adaptation. Finally, a summary of other efforts and processing required for efficient physical operation of a mobile radio communication system is given.

2.1 The Nature of Fading

If we regard the received radio waves as being composed of simple sinusoids with varying amplitudes, phases, and frequencies, and assume that all the parameters can be estimated, then the received power can be found as a function of the distance to the radio source [1, 15, 51]. The fading occurs when many sinusoids interfere, adding up to a varying resulting power at each point in space.

2.1.1 A Multipath Fading Experience

When sitting at the edge of a swimming pool, splashing your feet in the water, you immediately observe waves propagating out, in a circle, from the point where you sit. At first, the waves look smooth and even, almost like sinusoids. But then the edges of the pool begin to interfere with the waves, causing reflections of the waves. The reflections go out in an angle symmetrical to the incoming waves, with respect to the edge. When the incoming and outgoing waves collide, their heights are added, so that two peaks cause a peak with double the height, and two valleys double their depth. A peak and a valley will result in a cancellation, so that no resulting wave will be observed at that point in the pool.

Now consider the case when you want to transmit information to someone at the other side of the pool, only by causing waves to propagate to the receiver of the message. The agreed upon code is simple: Waves means "unhappy" or "hungry", and absence of waves means "happy" or "indifferent". Now, the receiver's foot, the detector, is unfortunately resting over the point that results in a canceled wave, so no message can be received until the receiving foot moves to a better location.

2.1.2 A Relevant Example

The pool-side example illustrates a simplified scenario of multipath fading. In real-life, the problem addressed is that different radio signals interfere, also with themselves, so that the received signal power varies as the radio receiver travels through the air. A typical scenario of multipath fading radio transmission is shown in Figure 2.1, and a typical resulting received power signal is shown in Figure 2.2.

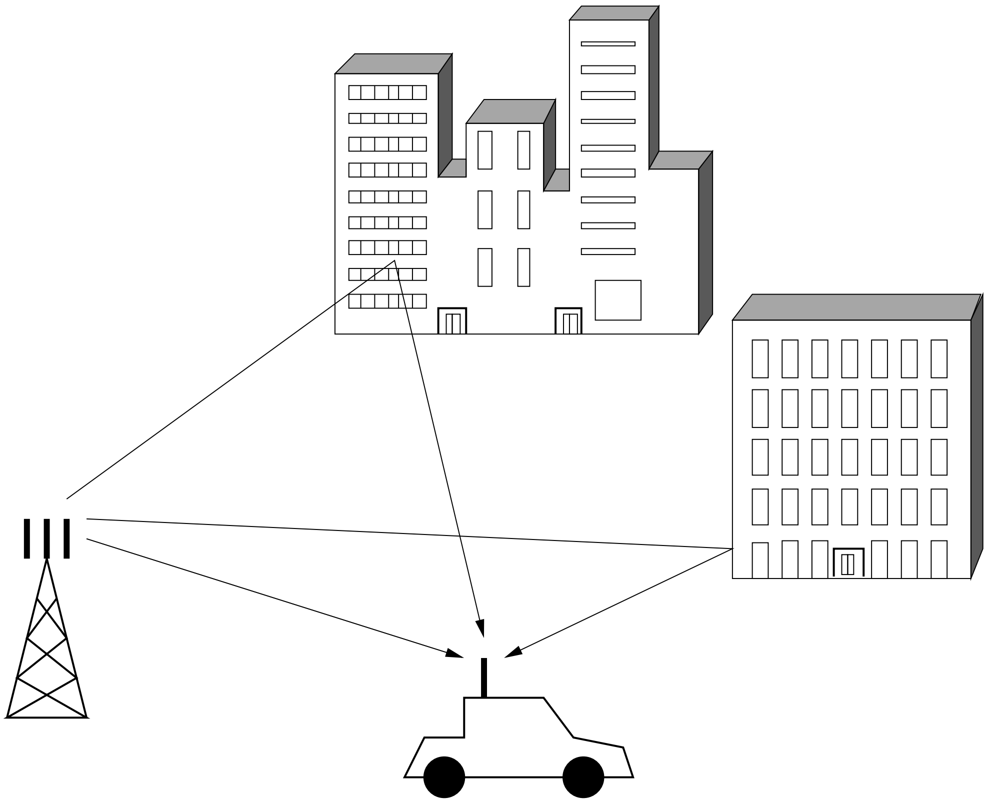

Figure 2.1: A vehicle with a mobile radio receiver experiences multipath fading when receiving delayed and phase shifted rays from the base station. Also note that the multipath signals will result in different Doppler frequencies, since the vehicle travels towards a scatterer, and away from the base station.

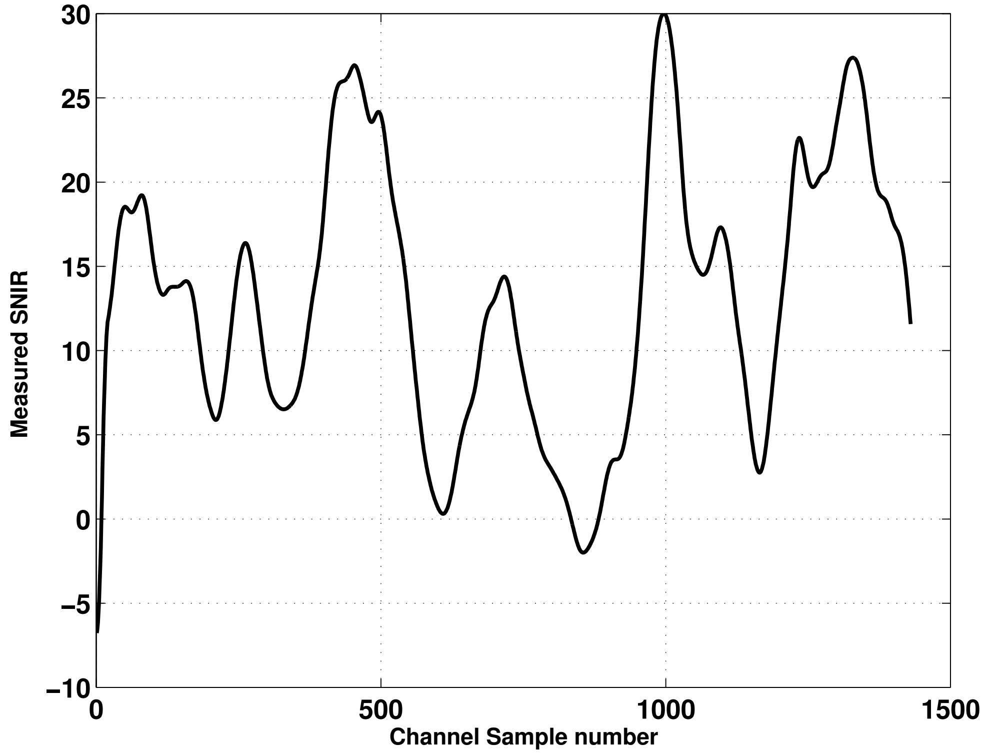

Figure 2.2: A typical experienced received power signal of 5 MHz bandwidth at 1800 MHz, moving at vehicle speed. The measurement contains 1430 power samples [15]. Each of the samples is calculated from an impulse response, based on a measured sequence of 700 signal samples, sampled at a rate of 6,4 MHz. This results in a channel power sampling frequency of about 9,1 kHz.

2.2 Can the Fading be Predicted?

It seems reasonable that the fading pattern should be predictable, since it basically results from sums of sinusoids of different (Doppler) frequencies, and different amplitudes, coming in from different directions. In [15] it is concluded that with reasonable approximations in the wave model of the channel, theoretically, a model stays valid for a time-window in the order of 100 ms, or a few wavelengths. From this it is concluded that adaptive models and predictors should be used for the purpose of modeling and predicting the fading pattern of the mobile radio channel. The channel is modeled by its sampled impulse response, where each tap is regarded as a discrete time-series. An acceptable prediction gain is achieved for ranges of up to half a wavelength, which at a carrier frequency of 1880 MHz and a mobile speed of 10 m/s (36 km/h) corresponds to 8 ms.

A mobile wireless channel can be modeled by its impulse response. Each tap in the impulse response represents a time-lagged sample of the transmitted signal. In Figure 2.3, we see that fairly accurate predictions can be achieved by methods based upon linear filtering of individual past channel taps [56].

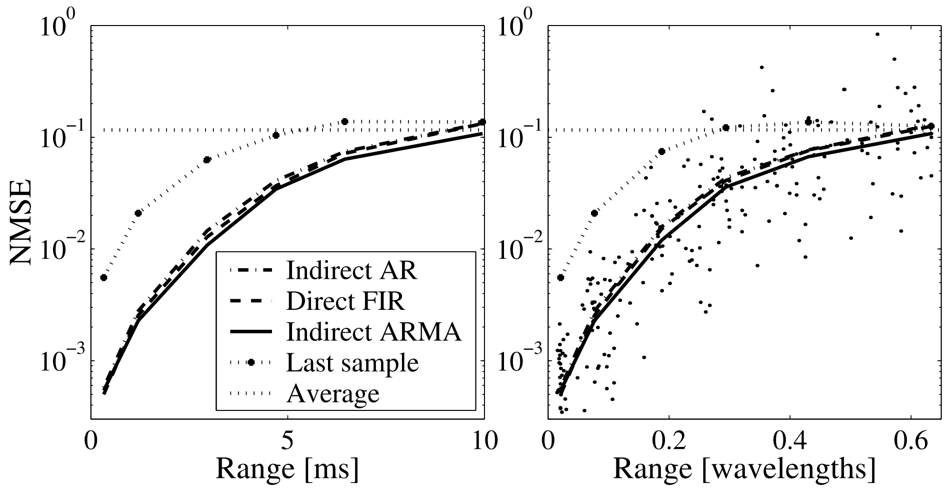

Figure 2.3: Figure from [56], showing the medians of the normalized prediction mean square error (NMSE) for the received power at 45 locations, predicted with different methods. The data that the plots are based upon, comes from measurements of real-life broad-band channels. The plots show that linear model based prediction of channel quality provides a much better estimate than average power and last sample estimates, and that fairly accurate predictions can be achieved for several milliseconds ahead in time.

One of the questions this thesis tries to answer, is how well we need the predictor to work in order to obtain a superior system performance as compared to a non-predictive approach.

2.3 Diversity or Adaptation?

One way to deal with the problems caused by multipath fading and interference, is to use long interleavers and channel coding [46, 51]. Interleaving can be regarded as a diversity technique that fights the effects of fading by spreading out the transmitted message over time, so that temporary bad conditions are compensated for by temporary good conditions. Interleaving simply re-orders the bits that will be transmitted, so that bursty error patterns are spread out over time, feeding the decoder with hopefully non-bursty error patterns. The temporary high error rates are spread out over time, so that, after the interleaver, the error rate becomes moderate over the whole interleaving interval. The resulting error rate can then hopefully be handled by the channel decoder, described next.

Channel coding adds controlled redundancy to the transmitted sequence of bits, reducing the number of allowed code-words that the decoder accepts. As a result, the decoder can, at a high probability, deduce when an error has occurred, since it will not recognize the received word as a valid code-word. Often, when the damage is not too great, the decoder can also make a correction and output the code-word that is closest to the received erroneous one. In order for this to work, the errors must not come in sudden bursts, which is common when the channel is subject to fading. To counteract the burstyness, interleavers are used.

If channel coding and interleaving are used as the only means to alleviate the problems related to fading channels, a trade-off will have to be done. Either we will have to use a very robust channel code that can cope with high error rates, so that occasional bad signaling conditions don't disrupt the data transmissions[1]. Or, we will have to tolerate more or less frequent errors in the decoded data by using a less robust channel code, resulting in higher layer re-transmissions, which also results in overhead as repeated transmissions of the same data.

Other ways of creating robustness against channel impairments, are frequency hopping and direct sequence spread spectrum, which both can be regarded as diversity techniques since they spread the information over a wider frequency spectrum than the transmitted information requires.

An alternative to the diversity approach, that avoids the mentioned trade-off, can be obtained utilizing knowledge of the channel state. Instead of fixing a level of overhead that can cope with worst-case conditions, we can let the error protection vary as the conditions vary. That is, the overhead is always adapted to the current conditions, avoiding both over-pessimistic channel coding when conditions are good, and resource wasting re-transmissions due to insufficient error protection.

The fading behavior of, and the possibility to estimate a wireless channel, is

[1]A robust channel code implies that a large amount of redundant information has to be constantly transmitted as coding overhead, since a worst-case scenario has to be assumed.

governed by its bandwidth, the exploited diversity, and, the amount and type of disturbances. Increased bandwidth and diversity results in less extreme variations. A GSM channel will experience higher variations than a W-CDMA channel. Also, utilization of receiver diversity, through for example polarization diversity, will help reduce the amount of variations.

2.4 Other Necessary Processing

In combination with the diversity techniques outlined above, we may need to use some processing to actually *improve* the signal strength experienced by the receiver. This improvement can be achieved through space-time processing of not only the received signals, but also the transmitted signals. The methods used for space-time processing include *spatial diversity* through multiple transmitter and receiver antennas, *temporal equalization* through combination of delayed signals, and *interference rejection* through identification and subtraction of interfering signals from the desired signal [38, 61, 65].

We must also recognize the necessity of deployment of improved hardware in the communicating radio devices, which reduces the amount of thermal and quantization noise, as well as other types of noise induced by the physical limitations of existing radio devices [31].

The results described in this thesis are obtained assuming ideal conditions concerning the temporal equalization. Channel simulations are carried out using additive white Gaussian noise (AWGN).

Chapter 3

Application Layer Considerations

Some issues at the higher layers in the communication protocol stack are discussed in this chapter. There are many different types of user information that can be transmitted over wireless channels in a future mobile communication system. They range from file transfers, to interactive multimedia communication, from short administrative control messages, such as geographical location information, to continuous transmission of measured data, such as critical medical or environmental probing information.

What can be distinguished from these different applications are the varying requirements they pose on the communication system used. Some applications require short end-to-end transmission delays, while others require high and efficient usage of raw system resources.

Within the 3rd Generation Partnership Project, a set of four different classes for Quality of Service over wireless, have been defined [60]. They are further described below.

3.1 Real-Time Applications

A real-time application is characterized by the required proximity in time between the creation of a message at the source to its reception at the destination. Real-time applications can be further subdivided into *conversational* and *streaming* applications. The delay and jitter requirements are higher on conversational applications, since they have requirements in both directions of the connection, so there is no reasonable way of buffering large amounts of data in order to absorb the channel variations.

A streaming application can be relatively insensitive to delay, but not to jitter. The transmission delay can be dealt with by just waiting until the data comes through, e.g. a video on demand transmission can always hide the delay by just letting the user wait for the movie to start. The jitter, however, has to be dealt with through buffering.

3.1.1 IP telephony

Conversational applications, such as telephony over IP networks, are characterized by small but frequently transmitted packets. A GSM speech encoder for example, generates a voice packet of 260 bits every 20 ms [47] resulting in a bit rate of 13 kbps. The receivers in such applications are usually not sensitive to occasional loss of information. First, the decoder is often capable of interpolating over a lost segment of data, and second, the listener (human) can also tolerate some loss, since human hearing can filter out short disturbances.

On the other hand, there is little tolerance against delay in the transmission. A delay higher than 200 ms is disturbing for a conversation, leading to the corresponding parties' speaking at the same time.

Internet communication lead to a high variety in transmission delays, since the path over which packets travel can change from one packet to the next, thus resulting in poor experienced quality.

3.1.2 Real Audio

A streaming real-time application, is the Real Audio multimedia streaming application. This normally uses the TCP protocol for transmission from the server, and relies on buffering at the client application to smooth out network jitter [53].

It can also utilize proxy servers that reside closer to the client than the original source server, in order to reduce both delay, jitter, and network load.

3.2 Non Real-Time Applications

The most obvious example of non real-time applications is a simple email program. It does not even require the sending host to be connected to a network, since the program can choose to send the email at a later occasion, when the host gets connected. Non real-time applications can be divided into *interactive* and *background* applications. Email and file transfer are typical background applications. An example of an interactive application is web-browsing.

3.3 Scalable Media

To reflect the future possible variety in user terminals and user interfaces, as well as a large variation in the achievable bandwidth over different networks, some media types allow the contents to be scalable, so that not all of the generated source code is necessary for decoding of the application data. This implies that, if the complete source information cannot be delivered to the user, should it be for network or terminal reasons, the most relevant part should be prioritized in order to give a good-enough service.

3.4 Actual versus Experienced Bandwidth

In general, there is a difference between the experienced and the actual data throughput. The experienced throughput reflects the illusion of, for example, page downloads while browsing the web. In practice, web pages contain a high level of redundant data, data which does not necessarily travel across the wireless link at each separate download. What this suggests is that redundant transmissions over wireless links should be avoided. Instead, a trade-off between the necessary computing power at the clients for cache handling and compression on one hand, and the usage of channel resources on the other, should be carried out.

Chapter 4

Transport and Network Layer Considerations

The thesis assumes that the network will be IP-based. This is not an obvious choice for mobile solutions over various physical links and networks, but it provides an attractive transparency for application and service developers.

There are two main transport protocols over IP networks, namely the User Datagram Protocol (UDP), and the Transmission Control Protocol (TCP). They offer quite different services for the higher layers. TCP is an end-to-end connection oriented protocol that can associate a packet to a certain session or stream in the transport layer, whereas UDP does not have these features, but rather a packet-by-packet context, that relies on the application layer to associate the content to a certain session.

These two different transport protocols are discussed in this chapter, with focus on the issue of transmission over error-prone links, such as a wireless mobile link.

4.1 TCP/IP

TCP is a reliable transport protocol for the Internet. TCP is designed to ensure that the transmitted data reaches its destination without error. It also controls the transmission rate, based on feedback signals from the destination host [2, 49, 55, 58].

The common reflection one makes about TCP in a wireless environment, is that TCP was designed for high-bandwidth, short delay, *congestion* limited networks, and is not well suited for high loss, high delay, *error* limited links. The meaning of this is that the flow control mechanisms in TCP react wrongly on the (absence of) feedback signals, when used over wireless links. The TCP sender assumes that packets are lost because the packet stream is subject to congestion, and reacts by re-transmitting the lost segments and reducing the transmission rate by resorting to slow start mode, as described below.

4.1.1 TCP Flow Control

TCP maintains a number of state variables in the sending and receiving hosts. One is the Maximum Segment Size (`MSS`), which is the maximum number of bytes that can be held in one packet. This number is negotiated between the two communicating parties and is regarded as a constant, but it can be re-negotiated.

The two parties also adjust a window variable each. The receiver's *advertised window* (`AW`, see Figure 4.1) is a value reflecting the amount of buffer space that the receiver has for the current connection. It may change during the connection. At the sender side, a window called the *congestion window* (`cwnd`), estimates how much data the sender can output on the network before expecting an acknowledgment (ACK) from the receiver. The `cwnd` grows with one segment for each received ACK, resulting in an exponential growth of the transmission rate. The sender, however, must never output more data on the network than the receiver's advertised window.

The sender's `cwnd` is increased in this fashion until it reaches the slow start threshold (`ssthresh`), where it shifts from *slow start* mode to *congestion avoidance* mode. In congestion avoidance a different strategy for increasing the window size is adopted. The window is then increased by one segment at every round-trip time (`RTT`), regardless of the ACKs (though some implementations approximate this by increasing the window by a function of `MSS` and the current window at each reception of an ACK), resulting in a linear growth of the transmission rate. The window keeps increasing in this fashion until either DupACKs (described below) are received, or the re-transmission timer times-out.

Duplicate Acknowledgments (DupACKs)

When a segment is lost and does not arrive at the receiver, but other (subsequent) segments do, duplicate ACKs are returned to the sender. When arriving at the sender they invoke a recovery mechanism to deal with the assumed loss. First the missing segment is re-transmitted, then `ssthresh`

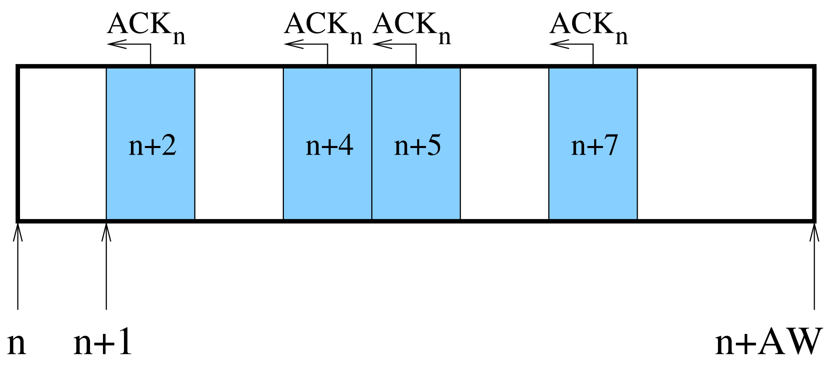

Figure 4.1: The receive TCP buffer. Segment n has been received and acknowledged, but segment number $n+1$ is missing, so for each arriving segment that has a higher sequence number than $n+1$, an ACK with segment number n, is transmitted to the sender. This indicates that segment $n+1$ is missing, but does not explicitly say anything about other already received segments. AW refers to the Advertised Window. When segment $n+1$ finally arrives, ACK_{n+2} will be sent, indicating that all segments up to $n+2$ have arrived, and that the next expected segment is $n+3$.

is reduced, and finally, the sender's window is reduced to `ssthresh`. Thus, TCP remains in congestion avoidance mode, but the increase in window size will now start from a new, lower, initial value `ssthresh`.

Re-transmission Time-Out (RTTO)

Another event that can generate a reduction in transmission rate, is the re-transmission timer time-out, which detects an ACK not being received within a reasonable time interval. This causes `ssthresh` to be reduced, but the sender window is also reset to its minimum, since it is more likely there is a congestion problem in the network. The sender thus reverts to slow start mode.

4.1.2 TCP Enhancements over Wireless

When crossing a narrow band error-prone link, where bandwidth efficiency is crucial for performance, many subsequent errors will cause re-transmission time-outs in the TCP sender. These time-outs will result in the lost segments being re-transmitted from the sender and each time, the TCP transmission rate is reduced to the minimum. This is not a desirable reaction in this narrow-band environment. We would like to avoid the repeated transmission of TCP segments, since they add to the bandwidth waste.

Several methods have been proposed to enhance the performance of TCP over error-prone wireless links [3, 6, 27, 41, 58]. Some require changes of the current implementations of TCP on either side, and some improvements can be done without such changes. Improvements that do not require changes at the end points, will in some way interfere with the philosophy of end-to-end connectivity. Different approaches to TCP enhancement methods are also compared in [21].

No changes to TCP implementations

Indirect TCP In this "split connection" approach [58], the TCP connection from the fixed network is terminated at the base station, and a separate connection is used between the base and the mobile station - not necessarily a TCP connection. The second connection is specially designed for the wireless conditions. This makes the TCP sender unaware of the wireless conditions, and a proxy server at the base station will play the role as the destination TCP. Another consequence is that TCP segments are ACKed even though they might not have reached the destination host.

Snoop Agent By letting a piece of software at the network layer, located at the base station and named *snoop agent* [3], "sniff" on the contents of the IP-packets, it can do some book-keeping and local caching of the passing TCP segments. The local cache is intended for local TCP re-transmissions. By not letting duplicate ACKs from the mobile host travel back to the TCP sender, and instead take care of the re-transmissions locally, one can avoid invoking the TCP back-off.

Both are examples of *Performance Enhancing Proxies* (PEP), on which work is ongoing within IETF [6].

Changes to existing TCP implementations

TCP SACK The receiver may be missing some segments in its receive window, while others have arrived, but out of order. This means that there are "holes" in the received data buffer, see Figure 4.1. The normal behavior of the receiver is to send duplicate ACKs for each out-of-order segment, so it is up to the sender to understand that a non-ACKed segment is lost, but it can only guess how many are missing. Different

TCP implementations deal with this differently [27]; Reno TCP re-transmits one missing segment per round-trip time (risking a long delay), and Tahoe TCP re-transmits several consecutive segments (risking re-transmission of already received segments). A selective ACK (SACK) [42] implementation of TCP, however, explicitly tells which contiguous blocks of segments have been received, substantially decreasing the risk of re-transmitting redundant segments or increasing the delay.

TCP Westwood In this new implementation of TCP, the idea is to keep an estimate of the throughput bandwidth at the sender, and use this as back-off after a loss is discovered, instead of resorting to slow-start [41].

4.2 RTP/UDP

For real-time conversational applications, the benefits related to TCP flow control, are not required. Nor are the delays from slow-start mechanisms acceptable. If a segment of the data is lost during transmission, there's no point in notifying the source, since the information will be outdated when it finally reaches the destination. Robustness has to be built into the application and into the source encoder, since a reliable service from the transport layer would be unacceptably slow.

The Real-time Transport Protocol (RTP) has been specified to meet the necessity for a flexible application-layer protocol for real-time interactive communication [52]. It usually relies on the User Datagram Protocol (UDP) for the transport, so no guarantees of delivery can be given from the transport layer. Its main contribution is sequence numbering, and framing of the data directly by the application. Together with the RTP Control Protocol (RTCP), the performance can be monitored and multiple parties introduced in an ongoing communication session.

4.3 IntServ and DiffServ

Two somewhat different approaches to control the service quality given by an internetwork have been suggested, namely IntServ [7], and DiffServ [5]. They stand for Integrated Services and Differentiated Services, respectively. The main difference between them is in the way they control the service given. Roughly, IntServ handles the service level by means of admission control and reserved bandwidths for packet streams, while DiffServ handles the service quality in a packet-by-packet manner, using tags in the packet headers [34].

4.3.1 Integrated Services

This approach to QoS provisioning is mainly application driven, meaning that the application negotiates with the network in order to find an appropriate service level. A reservation for a certain service level is set up along the transmission path. From the application point of view, this is an attractive approach, since the application gets information of what service levels the network can handle along the transmission path, and is therefore able to adapt the source coding accordingly.

4.3.2 Differentiated Services

Here, the network handles the traffic forwarding, without interference from the application. The application only labels a packet with the desired service level, so no negotiation takes place. An advantage of this approach is its flexibility; since no promises are given, none can be broken. Each node along the network path can handle the packet and its requested service in the way that suits the current link and the local network status.

4.4 IP Header Compression

For applications where many packets are transmitted along the same path, with relatively small payload, such as voice over IP, a major burden is the large overhead related to the packet headers. For the RTP/UDP/IP stack, the headers comprise of at least 40 bytes for each packet. If a speech packet payload consists of 15-20 bytes, the overhead is larger than the actual data transmitted. Substantial increase in efficiency can be achieved if the header can be reduced to a minimum.

At the time of writing this thesis, standardization work is under way regarding header compression. The work was initiated at Ericsson Erisoft, under the name of ROCCO (RObust Checksum-based header COmpression), and is now a part of IETF (Internet Engineering Task Force) standardization, under the name of ROHC (RObust Header Compression). At its best, ROHC can compress RTP/UDP/IP headers to an average size of slightly more than one byte!

The following sections are devoted to explaining how header compression is achieved, primarily based on the work done by the ROHC group [35].

4.4.1 Redundancy in Packet Headers

The fundamental insight that makes header compression possible is that the headers contain highly redundant information, both within themselves, and between consecutive packets. For example, the packet header field containing information about the length of the packet, is completely redundant, since this information can be obtained from the link layer. Further, the IPv4 packet ID number field is often implemented as an incremental counter, so if the ID for one packet is known, then the following ID numbers can be calculated from that.

From these examples one can deduce that a thorough examination of the different header fields and their variability can lead to a substantial reduction of the amount of data that has to be transmitted over narrow-band (wireless) links.

4.4.2 Header Compression Principles

The main principle used in header compression, is the maintenance of header contexts on each side of the narrow-band link, and calculation and transmission of so-called *deltas*, see Figure 4.2. The deltas contain, as the name suggests, differential information, which is the change of the header from the previous state to the current. A state is kept at both the compressor and the decompressor, and the idea is to maintain the same states at both sides, by transmitting as few bits as possible.

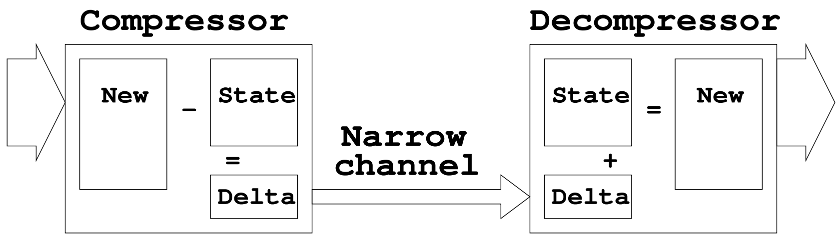

Figure 4.2: Basic header compressor and decompressor idea. The deltas are calculated from the new header and the previous header state, which is kept in both the compressor and the decompressor. The decompressed header is found by doing the inverse operation at the decompressor.

4.4.3 Header Context Updates and Repairs

When a new packet arrives at the compressor (transmitter), an update, the delta, is calculated so that the de-compressor (receiver) can figure out what the new header looks like. When an error occurs, the receiver has to detect it. Then, there must be some mechanism so that both wireless ends can synchronize their header contents. The repair mechanism could be "built-in" in the compressed header, so that the de-compressor detects the error at an early stage, and, based on the out-of-order compressed packet header contents[1], makes a good guess, and updates the new header. If the guess is correct, then the IP checksum adds up to the correct value, and the de-compressor has recovered from the error, without the need of re-transmissions and complete context recovery transmissions.

In a less robust setup, when the recovery is not "built-in" in the compressed header, the error will not be detected until the de-compressor calculates the IP checksum, based on the new compressed header and the previously stored context. Since there is an error, and we are not able to localize the cause, the de-compressor needs a complete context transmission from the compressor.

4.4.4 Compression Profiles

The efforts of the ROHC group has been focused on the RTP/UDP/IP stack, since an analysis of the involved header fields is required in order to achieve an efficient compression. ROHC does not specify a certain algorithm for the compression, but instead provides a guideline for the considerations that need to be made to design an efficient compression *profile* for a certain setup. ROHC makes use of a set of compression profiles, which are not specified by ROHC, but are to be designed by the application designers. The idea is that there should be a number of profiles to choose from, depending on the application in use, processing power of the terminal, required performance, IP version, and other such preferences. There are however a number of requirements on a profile. It must specify:

- Length of and coverage of the header checksum
- Needed context information for (de)compression
- Procedures for (de)compression
- How compression is initiated

[1] It is possible to detect how many packets have been missed.

- Context repair mechanisms

4.5 IP Payload Compression

There have been suggestions for an IP-layer compression of payloads, e.g. IPComp [54]. However, there are some drawbacks of such an approach, making it impractical for real use. First, the payload may already be compressed at the application itself, and then certainly by a more specific and efficient algorithm than a generic one that would be the case in IPComp. Second, the payload may be encrypted, and encrypted data looks random, which makes compression, that works by removing redundancy, rather inefficient.

These reflections suggest that compression applied at the source of the data, and not in the network, would be more efficient than a generic network-based solution.

4.6 Mobility Management - Network or Link Issue?

Mobile IP defines means for mobility management in IP networks [48]. Hosts are allowed to roam in remote networks where they are regarded as guests. The roaming host (Mobile Node)[2] announces its home network address to the remote gateway (Foreign Agent), which responds by sending the information to the host's home network gateway (Home Agent). Now the home network gateway can create a tunnel to the remote network, so that the host can be reached at the remote network. The tunnel is simply a new IP-packet, having the remote gateway as destination, in which the original IP-packet has been included. The original IP-packet can then be unpacked by the remote gateway, and delivered to the roaming host.

This approach requires that all packets having a roaming host as destination, have to pass by the home network gateway, before they can be accurately addressed and sent in the right direction. This is of course a drawback that increases the transmission delay. Variations to this technique have also been proposed, for example, the home gateway can send the information about the roaming host's current location to the host (Correspondent Node) that is currently transmitting data to the roaming host, letting the data take a more direct path to the destination host. Work on this, in combination with route optimization approaches for IPv6 is under

[2]The names in parenthesis refer to the entities defined in [48].

way. Drawback: When the roaming host moves to a new network, the previous care-of address will be inaccurate.

In a wireless and really mobile world this seems a bit stiff. Would this approach be able to handle such things as hand-over between base-stations? Should there be a border between the IP-based transport network, and the link-specific access network, such as a GSM or UMTS network, or will IP become truly mobile? These issues are also discussed in [43].

4.7 Summary

Several aspects concerning the transport and network layers have been outlined in this chapter, some of which have been taken into account in Chapter 6. The primary issues addressed in that chapter are quality of service (QoS) and spectral efficiency. In this chapter, both issues have been discussed: The QoS can be improved by letting the lower layers know about the applications running on the hosts, but also by transport layer enhancements, such as PEPs, mentioned in Section 4.1.2. Spectrum efficiency can be improved by different compression approaches, discussed in this chapter. But spectral efficiency can also be improved by, again, letting the lower layers know what level of service that is required by the application, so that a resource scheduler can make a spectrum efficient decision, still meeting the application requirements.

Chapter 5

Link and Physical Layer Considerations

The thesis addresses digital communication over mobile wireless systems, with some emphasis on *mobile* and *wireless*. A wireless communication link can be fixed, such as a microwave communication link between two corporate buildings, or even as an optical link between two buildings. In that scenario, there are no tight limitations imposed by the system on the transmitted power. Nor are the transmissions subject to any disturbances from mobility of the communicating stations (however, optical links need compensation for motion in sky-scrapers, due to uneven heating or wind).

Another scenario is the *nomadic* data communication. Here, a host can roam into a new network and connect either through a wire, or wirelessly, usually accessing through a local area network (LAN). Link layer difficulties are limited, since this type of connections do not support motion of terminals. This means that channels can be considered as static and no fast fading is present, allowing for high transmission rates.

In this chapter, however, issues relating the lower layers in the protocol stack to *wireless* and *mobile* communications, where terminals move at a relatively high speed, will be discussed. The focus is mainly on link adaptation, link layer ARQ (Automatic Repeat reQuest) and prediction of the radio channel's quality.

5.1 Modulation

Modulation is the process of translating digital information into physical wave-forms that can be transmitted over a radio channel, or on a copper wire. The process has to be reversible, so that a receiver can detect the signal and estimate the transmitted information. There are many ways for modulating a digital signal onto a radio channel. One way is through changing the amplitude of the radio signal, *Amplitude Modulation* or AM. Another way is through changing the phase of the radio signal, *Phase Modulation* or PM. These are only two examples of how digital information can be represented for transmission, and probably, all methods have not yet been invented. In traditional digital communication systems, either PM or AM are used, or a combination of them, namely QAM (Quadrature AM), that uses amplitude information in two orthogonal phase directions.

The modulation alphabet complexity determines the number of bits that can be transmitted within one modulated symbol. The more complex the alphabet, the narrower the gap between the different symbols in the decision space, the higher the probability for an error in the detection of the symbol. To compensate for a more complex modulation alphabet in order to maintain a constant error rate, either the transmission rate has to be reduced by transmitting the same symbol for a longer time, or, the transmission power has to be increased.

5.2 Channel Coding

In order to cope with unavoidable errors in the wireless transmission, channel coding is used. The encoder adds redundancy to the transmitted bit stream, so that the decoder is able to deduce when an error in the received bit stream has occurred, and even make a correction if the error is not too extensive. The more redundancy that is added to the bit stream, the more robust the coding, and, the more errors can be corrected (to a certain limit). The increased robustness is paid for by a reduction in the efficient bit rate, since part of the allocated bandwidth has to be used for the added redundancy. A central term in this context is the *code rate*, R_c, which is defined as the ratio between the original bit-rate and the bit-rate after the encoder [46, 51, 66].

Channel codes are subdivided into *block codes* and *convolutional codes*. They can also be further subdivided into *systematic* and *non-systematic* codes. The different types differ both in performance and method of implementation.

A block code acts on data in mutually independent blocks, as the name suggests. From a binary block of k data bits, a new block of n ($n \geq k$) bits is calculated and is referred to as a *code word*. In the case of a *systematic* code, the original k bits are preserved in the code word, and the $n-k$ parity bits are appended, to build the code word. In this case, the code rate, R_c, would be k/n.

In a convolutional code, there is no such independence between different blocks of data. The data bits are convolved with one or more generator polynomials, depending on the code rate. The original bit stream is passed through one or more shift registers, that serve as sources for the convolution operations that result in the coded bit stream, see Figure 5.1.

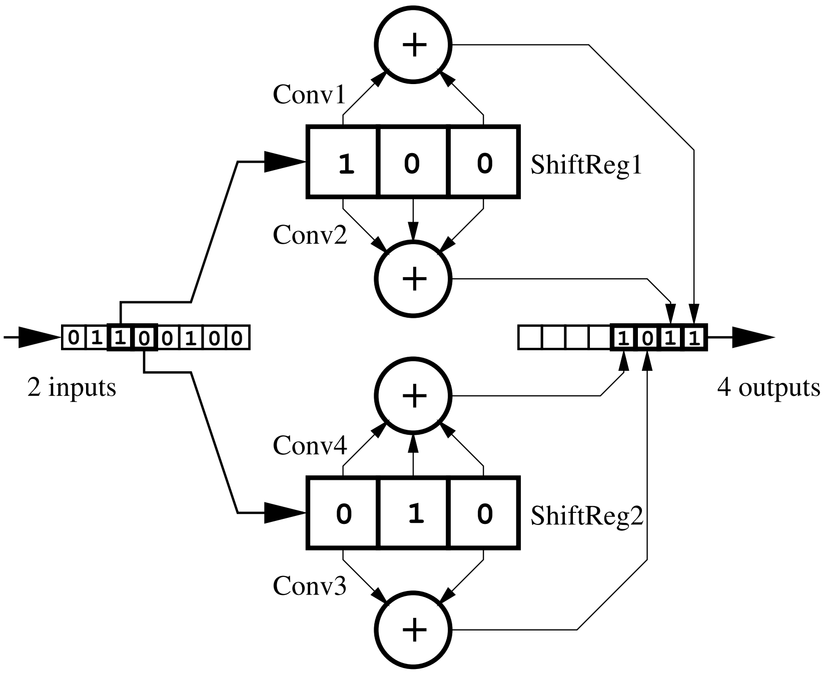

Figure 5.1: An example of a convolutional encoder with two input bits and four output bits, resulting in a rate $R_c = \frac{2}{4} = \frac{1}{2}$. The two different input bits are convolved modulo-2, together with the two previous bits, with the same generator polynomials, namely 101_2 and 111_2. (A "1" in the generator polynomial means a connection to the modulo-2 adder.)

5.3 Channel Prediction

Why try to battle the effects of fading by using averaging techniques, if it is possible to predict the channel and thus always use it at its capacity? Averaging techniques, such as interleaving, coding, frequency hopping, and wide-band spreading, have been discussed in Section 2.3. Here we focus on adaptive approaches, such as Hybrid ARQ [22] and adaptive modulation [11, 62], leading to channel predictive scheduling, which will be further described in Chapter 6.

5.3.1 How Can We Use Channel Predictions?

The question seeks an answer to what we would do if one knew what the future looked like. A mobile radio is allowed to know how the transmission quality will vary within a few milliseconds of the nearest future. How should this knowledge be used? The impact of a temporary bad channel on the bit error rate when transmitting digital information can be seen in Figure 5.2. Here, the mobile is transmitting with a constant modulation while the channel experiences several fading dips.

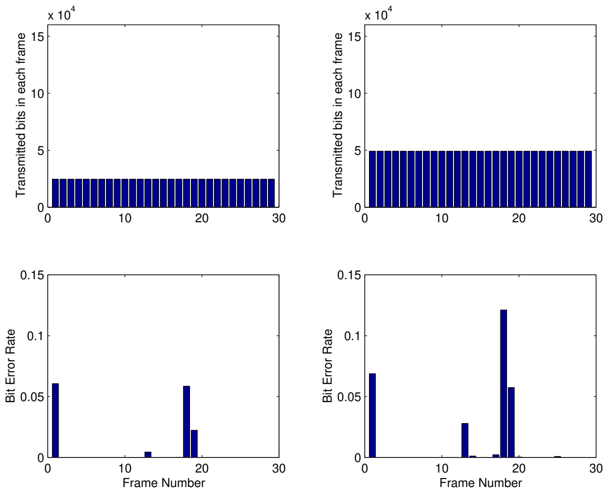

Figure 5.2: Constant BPSK (left) and QPSK (right) modulation and resulting error bursts when the channel experiences fading dips, especially around frame number 18. The two top graphs show the number of transmitted bits in each time-frame, whereas the two lower graphs show the bit error rates in each frame. The corresponding channel time variability is depicted in Figure 5.3, where one frame corresponds to 48 time-slots, or 5ms.

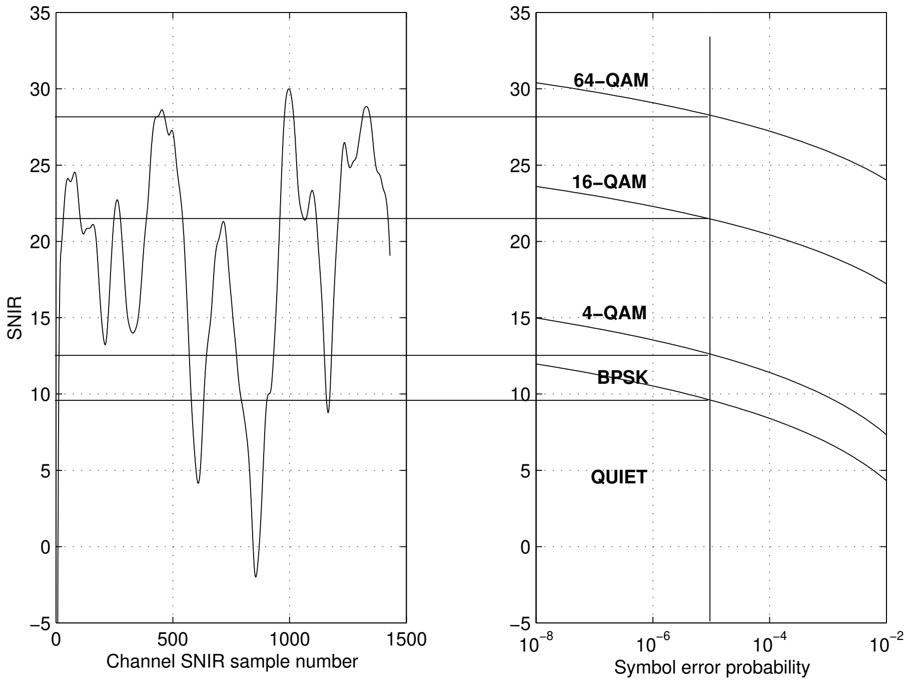

Figure 5.3: SNIR trend and modulation level related to the symbol error probability. On the right hand side we see how a chosen target error rate (10^{-5}) translates into a modulation level for a given SNIR.

A first obvious answer to the question asked above is to let the mobiles adapt to the changing channel quality by aiming at a target bit error rate (BER), or frame error rate (FER), through variation of the modulation alphabet [11, 36, 45, 62, 63]. In Figure 5.3, a simple illustration of this idea is given. The left hand diagram illustrates the Symbol-to-Noise-and-Interference Ratio (SNIR) variation of a typical broad-band channel, while the right hand part illustrates how the level of modulation can be selected for a pre-specified symbol error probability. The target symbol error rate is set to $1/10^5$ and the allowed modulation formats can be found as a function of the current SNIR, which varies with time. Here 64-QAM could be used as the maximum modulation level, thus transmitting six bits per symbol when the channel is at its best. When the channel degrades, lower powers of two are used with BPSK being the lowest level. As an example we note that for an SNIR $\geq 22dB$ we can transmit during 150 time-slots (time-slot 390 to time-slot 540) with a modulation level of 16-QAM at a symbol

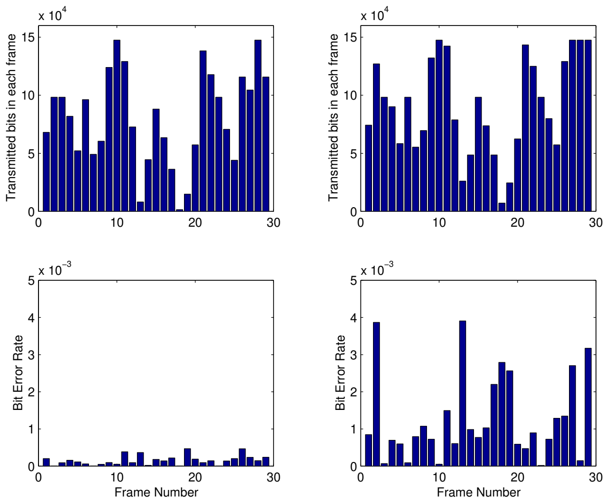

Figure 5.4: The bar plots show the result of applying the same channel impairments to a mobile, using adaptive modulation with two different target symbol error rates (10^{-3} on the left hand side, and 10^{-2} on the right). The two top graphs show the resulting number of transmitted bits for each time frame, which vary, since different modulation alphabets result in different bit rates for a constant symbol rate. The two bottom graphs show the bit error rates, which are considerably lower than the ones in Figure 5.2, where the modulations were static. Note that the range of the y-axes of the bit error rate graphs are smaller than those in Figure 5.2.

error probability of $P_M \leq 10^{-5}$. The results of two simulations with target symbol error rates [1] of 10^{-3} (left), and 10^{-2} (right), are given in Figure 5.4. Here we can see that the bit error rates are much lower than in Figure 5.2, and relatively constant, whereas the transmission rates vary according to the varying power of the received signal. The resulting throughput is defined as the number of transmitted bits in each frame. The bit errors are not subtracted in the definition of throughput that is commonly used, and is used here. This is a choice based on the fact that different applications require different link layer services, some of which can accept occasional errors in the transmitted data, and some of which cannot.

A second step would be to introduce adaptive coding rates, so that the error protection can be fine-tuned within each modulation interval [20, 29, 30]. This also gives a more fine-grained resolution of the control variable (namely

[1] The symbol error rate is approximately equal to the bit error rate in these examples, since at these low BER levels, the most dominant error pattern is one bit per symbol

the number of information bits per transmitted symbol), if we regard the problem as a control problem.

In a third step, assuming that all channel predictions are known in a centralized processing unit, we could use them for concurrent resource allocation in a shared physical medium. This is what is meant by the term *scheduling*, which will be dealt with in the next chapter.

5.3.2 Prediction Quality

The channel quality can be predicted, but the prediction performance decays rapidly with the prediction horizon, see Figure 2.3 on page 7. Prediction also becomes harder for fast moving mobiles than for slowly moving ones. This is due to the fact that prediction performance of multipath fading, is a function of the position of the receiver. When a mobile moves fast, its environment changes rapidly. Fortunately, a majority of the mobile users will not be moving at higher speeds, since they will be walking or driving in an urban area. For higher speeds, such as when driving down a freeway, or when traveling by train, special solutions can be accomplished, since the mobiles will travel along known paths, and thus have a low degree of randomness.

Another issue is to predict the interference from other users. Having no knowledge about the whereabouts and future transmission plans of a user, it will be very hard to estimate the interference in a packet based system. In a centralized approach, however, there is knowledge in the system about location, and, even transmission schedules, which could be exploited in order to take the generated interference into account.

5.4 Power Control

Power control aims at giving all users a fair level of radio power to attain pre-specified QoS levels, under the constraint of keeping interference at a minimum, or throughput at a maximum.

A basic relation in wireless digital communication, is that for a given bandwidth and a given ratio between signal power and noise, a certain transmission bit rate can be achieved at a certain bit error probability. Increased power enables a higher transmission bit rate at the same error probability, or a lower error rate at the same transmission bit rate.

If one mobile experiences a high rate of errors, it will try to compensate by increasing its transmission power. Apart from hopefully reducing its own error rate, the increased transmission power will also lead to an increased level of interference for other stations operating at the same frequency, consequently increasing their error rates. A compensation for this would be achieved if the affected stations also increase their transmission powers. It is easily seen that this "cocktail party" effect is catastrophic for a mobile system, leading to every-one increasing their voices in order to be heard.

Power control has to be handled centrally in order to avoid the cocktail party effect. The base station can measure the received power from the mobiles and thus demand an increase or decrease of their power. The mobiles also have to report their experienced received power to the base station, so it can adjust its transmission power.

Another issue with power control, related to the current work, is that *fast* power control counteracts the effects of fading, but at the price of generating a higher amount of interference in the system. Moreover, it would counteract our purpose of exploiting the variations for resource allocation and multiplexing, described in Chapter 6.

5.5 Adaptive Modulation and Coding

In traditional digital communication systems, the modulation is designed to cope with channel variations in a worst-case manner, that is, the system contains so much overhead that it can cope with the worst-case scenarios and still deliver an error rate below a specified limit. For wireless systems this implies the use of a simple modulation scheme, and a complex error-correcting code since a minimum SNIR cannot be guaranteed. When the coding fails to compensate for temporary bad conditions, higher layers in the protocol will ensure that the information is correctly and completely transmitted, if required, by requesting a re-transmission of the erroneous data. We wish to avoid all of this by adapting our demands on the channel as it varies. By changing the modulation format as the channel SNIR varies, through adaptive modulation, we accomplish a more stable performance with less re-transmissions [11, 36, 45, 62, 63]. In an adaptive modulation system, the symbol alphabet can be varied from one symbol to the next. It is, however, not practical to change the symbol alphabet too often, since the receiver either has to be notified of the change, or it has to be able to estimate the symbol alphabet currently in use. The rate at which we are allowed to change the modulation alphabet should on one hand reflect the added complexity of varying the modulation, including receiver setup and signaling overhead, and on the other hand, the gain in having a highly flexible modulation.

The modulation could be chosen so that it meets a required target error rate, as in Figure 5.3, dictated by the application that is currently using the link. For instance, a speech application does not require as high protection as a data file transfer.

5.6 Link Level ARQ

It is far from obvious whether or not the link layer should be non-intelligent and refrain from delivering the data without error. When reliable transport protocols, such as TCP, exist on top of the link layer, it is tempting to let them take care of transmission control issues. Traditionally, what the link layer does at the receiver, is to check whether the received data is correct or not, and if not, simply drop the frame. However, there is much to gain from a reliable link layer protocol. The link layer only handles the connection between two neighboring nodes in a network, and does not interfere with the end-to-end transmission control, which is handled by TCP. Therefore we could allow the link layer to have its own ARQ system, through feedback of ACK or NAK signals. So instead of just dropping an erroneous frame, it would send back a NAK and expect the transmitter to re-transmit the same data [33]. Alternatively, if Hybrid ARQ (see below, and [22]) is used, an incremental piece of data which can be combined with the already received frame, is transmitted, yielding a lower code rate for each re-transmission.

The obvious advantage of this approach, compared to transport layer ARQ, is the finer data granularity at this layer, and also the proximity in the network between the communicating nodes, leading to shorter response times. These features imply faster and more efficient re-transmissions, due to the shorter latencies between the communicating nodes and the smaller pieces of data that have to be re-transmitted.

The limitation on an ARQ system is imposed by the acceptable delay, which is reflected in the maximum number of transmission attempts, before regarding the data as outdated, or obsolete.

5.6.1 Hybrid ARQ

In a traditional ARQ system, an error is responded to with a re-transmission of at least[2] the entire lost data frame. This limits the system to transmission rates of 1, 1/2, 1/3, and so on, since each re-transmission does not provide

[2]Go-back-n ARQ systems usually result in more frames than the erroneous one being re-transmitted.

any new information, it only repeats what was previously sent. This is avoided in a hybrid ARQ system [14, 23, 25, 26]. To start with, the data stream is encoded using a low rate convolutional code, that, through tail-biting or zero-tail [22], results in a block-wise[3] encoded stream. Then, each block is punctured to several versions of increasingly higher rates, each version being a subset of the one with the lower rate. By doing this, it is possible to do re-transmissions using a new, incrementally redundant piece of data for each re-transmission, resulting in a lower code rate for each transmission, but with a higher coding rate resolution than a traditional ARQ system.

Assuming that we have a static channel, the Hybrid ARQ approach will automatically adapt to the code rate that is required for that channel, only with the cost of an increased delay due to feedback latencies.

5.6.2 Data Granularity

A trade-off that needs to be made at the link layer, is the size of data-units that can be resolved by the ARQ system. For each piece of data at the highest resolution, there has to be some control overhead, such as a header and CRC (cyclic redundancy check), to allow the receiving end to point out to the transmitter that a particular data-unit has been erroneously received. So, the amount of overhead in the normal[4] case, increases with increasing resolution. On the other hand, the amount of data that has to travel more than once over the air interface, in the case of a link layer re-transmission, decreases with increasing resolution, leading to a lower error-related overhead.

5.7 Summary

There are many possible improvements that can be introduced in the lower layers of the communication protocol stack to cope with future demands on throughput and bandwidth efficiency. The three main issues that will have a considerable impact on the performance have in this chapter been identified to be

- Channel prediction (range and accuracy)
- Adaptive modulation (number of levels and rate of change)

[3]In order to generate a decodable block, the remaining states in the encoder have to be brought back to the zero state (or another known state) by injecting tail-bits to the original data stream.

[4]With *normal* is meant the case when no re-transmissions are performed.

- Link layer adaptive ARQ (type and packet granularity)

In Figure 5.5 some link layer features and functionalities are depicted, and also their relations. Prediction of the channel quality depends on the type of channel that is used. Its accuracy is also very much dependent on the time range that the predictor should work in. The modulation levels that can be used depend on what kind of mobility and mobile services that should be supported. For example, it is desirable for fast moving terminals to change modulation level more often than the ones that experience a static channel. The prediction accuracy (or prediction error variance) affects the expected error performance of the wireless transmissions. The acceptable prediction accuracy also influences the choice of prediction range. This, together with the ARQ granularity, dictates the amount of overhead that will result from an ARQ system. The prediction range and its time resolution, along with the desired update rate of the modulation alphabet, determine the amount of signaling overhead required by the scheduling. The minimal desired prediction range is also affected by computational delays of the scheduling and prediction schemes. Also, the ARQ granularity controls the amount of resulting overhead, depending on if many errors are expected or not.

Other aspects of scheduling will be further discussed in the following chapter.

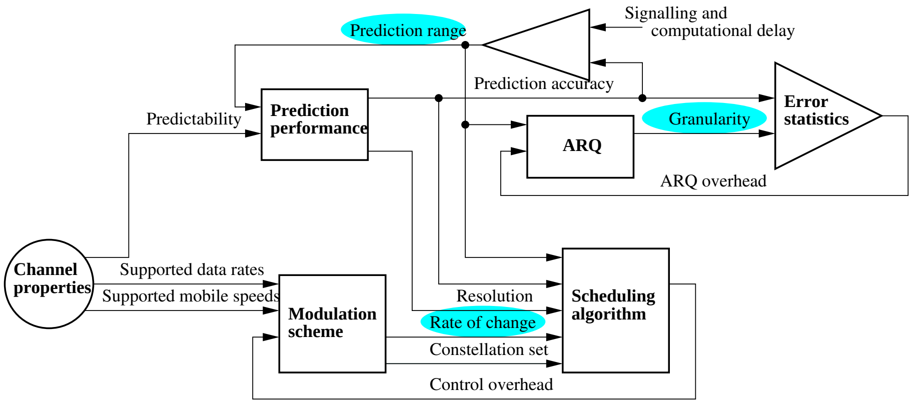

Figure 5.5: Different link layer features affect the performance of other features, some of which can be chosen as design parameters. The highlighted parameters decide the rate at which the modulation should be changed, the granularity of the ARQ system, and the range at which the predictor should work. The set of system parameters should be chosen so that the amount of overhead is minimized, while the required link layer services can be provided.

Chapter 6

Scheduling

We will study a broad-band, down-link dominated, time-multiplexed wireless mobile system, capable of dynamically mixing different types of traffic over fast fading channels. The purpose is, of course, to have an interesting and challenging topic of research, in an area too far-fetched for the industry to be of high priority at the present time [1]. The purpose is also to build a vertical knowledge in communication systems in general, that is, a knowledge across traditional technological borders that, for example, the OSI reference model dictates.

The preceding chapters have outlined some considerations to keep in mind at the different layers in the OSI reference model. For the continuation of the thesis, some of those considerations will be further dealt with, in terms of proposed and tested solutions. The proposed solutions are based on (adaptive) *scheduling*, since many aspects can be incorporated into the same level of abstraction: *Allocation of wireless resources based on their availability, and application requirements.*

Adaptivity is crucial to obtain spectral efficiency in future mobile wireless communication systems. Real-time predictive scheduling is one adaptive scheme that could provide improved performance. In order to achieve good scheduling performance, we need accurate long-term channel predictions. If such predictions cannot be made, we will fall back to non-predictive adaptive modulation or link adaptation. To achieve useful results from a scheduler, it is also necessary to have access to the different requirements set for different traffic classes.

[1] This has, however, changed since HSDPA (High Speed Downlink Packet Access) was introduced as an evolution for UTRAN by 3GPP [59]

6.1 Motivations for the Use of Scheduling

Efficient use of the available bandwidth and fulfillment of QoS requirements from higher layers can be accomplished by adapting the QoS demands to the current channel quality.

In a case where no fading is present, it would still be useful to schedule transmissions with respect to the different traffic classes that are served by the system. This is the case in, for example, ATM switches, where different services are given to cells belonging to different flows. The scheduling under these circumstances deals with the problem of allocating a known (fixed) resource to different traffic types, and to distribute the allocated resources among different flows within each traffic type [10].

6.1.1 Motivation 1: Improving Spectrum Efficiency

The channel quality varies substantially over time, due to radio interference and the mobility of the radio stations. Different types of fading will cause, at least temporarily, bad channel conditions. *Slow fading* can be counteracted by controlling radio transmitter power, or performing hand-over to another base station. The remedy against *fast fading*, however, is traditionally different types of channel coding and interleaving. The channel coding, which is often over-pessimistic, generates substantial overhead to the wireless system, which in turn wastes precious bandwidth. By using scheduling the available spectrum can be used more efficiently.

6.1.2 Motivation 2: Channel Prediction Works

A central component in the scheduling approach is the channel predictor. It is possible to predict the received power variations quite accurately several milliseconds into the future, even for fast moving vehicular users, see Section 2.2. Having these predictions, one per radio link, they can be used together with a target error rate to assign the modulation complexity, as described in Section 5.3.1, for planning of the transmissions to the different mobile hosts. In this way we increase our chances of transferring the data across the wireless link without error and at a high rate, thus increasing the system throughput, and the spectrum efficiency.

On one hand, the longer the time-frame the scheduler works on, the more optimized the allocation will be. On the other hand, the further into the future we look, the harder it is to make a correct and accurate prediction of the channel quality. So, in a real system, a trade-off between prediction horizon and scheduling gain, has to be made. The performance for some different link-layer strategies, with unreliable channel predictions, are evaluated in Chapter 7 and in [19, 23].

6.1.3 Motivation 3: Fulfilling Quality of Service Guarantees

Another issue for future data communication over wireless links, is quality of service (QoS). Although there is no agreed-upon definition of what QoS really is, it most certainly is better met by increased throughput, reduced error rates, and decreased delays. The bandwidth over the wireless channel is limited, so it has to be utilized efficiently. A way of doing so is to associate each data flow with an economic value, and to give priority to the higher valued flows. This value assignment is not trivial, but once it exists, it can be incorporated into the scheduling so that the scheduler tries to maximize the value of the transmission, and hopefully, provide the required QoS.

Alternatively, different flows can receive different services, simply based on their traffic class, as with ATM networks [28, 44].

By matching the higher protocol layer requirements with the physical constraints of the radio channel, we hope to find a time-slot allocation that efficiently utilizes the available spectrum, and maximizes the value of the transmission.

The knowledge of current and future channel conditions is accomplished through channel prediction, described in Section 5.3, and through centralized calculation of future scheduled transmissions, which will be described in this chapter.

The resources that are scheduled for different users, based upon their demands and the availability of the required resources, can be partitioned by means of time multiplexing (TDMA), or frequency multiplexing (FDMA), or even both.

6.2 System Overview

The system we are considering is regarded as a part of the Internet, not only a wireless extension of it. By this we mean that base-stations should have network layer routing functionalities, being able to store and forward packets depending on routing table entries. However, this particular network node, the base-station, has some extra abilities not expected from an ordinary router. Apart from forwarding packets to and from mobile terminals, the node should offer performance enhancing proxy (PEP, see Section 4.1.2) services, such as involving for example, a Snoop Agent, or a split TCP connection[2]. The node should also be able to distinguish between different traffic classes, and map them to a particular service level. The service levels that can be provisioned are influenced by the physical channel properties. These are continuously estimated for active sessions, and are forecasted with channel predictors. Also, at the link layer, the node should have an ARQ-system able to perform fast re-transmissions of link layer data frames.

6.2.1 Assumptions

The main assumption made, is that the traffic is down-link dominated, meaning that more data flows in the direction from the network to the mobile terminal, than in the other direction. The wireless access scheme reflects this assumption in that the down-link transmission is the one that is dynamically scheduled, and optimized, in order to utilize the wireless resources as efficiently as possible.

The studied system utilizes a symbol transmission rate of approximately 5 million symbols per second, to be compatible with 3G frequency planning.

In our system we have chosen to carry out the multiplexing of users on the shared channel, in the time domain. This approach is generally called TDMA (Time Division Multiple Access), as opposed to other multiplexing methods, such as FDMA (Frequency DMA) and CDMA (Code DMA). This assumption is only a first step towards a more general multiplexing, depending on the partitioning of the wireless channel resources.

Time has been divided into frames and time-slots. One frame was chosen to be approximately $5ms$, reflecting an initial assumption that channel quality could be fairly accurately predicted for $10ms$ ahead in time[3]. The frame timing of $5ms$ was chosen so that the scheduler should be able to work with batches of channel predictions, see Figure 6.1. Each time frame is then divided into 49 time-slots. The time-slots have been chosen to be $109\mu s$ long, the rate at which the available channel estimations are updated. Each channel estimation gives rise to a new time-slot. These time-slots are the resource units that the scheduler works with. The channel data used for evaluating the scheduler were recorded in Kista, a suburb outside Stockholm, and they are more thoroughly described in [15, 56].

[2] The interaction between such performance enhancing proxies and the present system are not studied in this thesis, but they are a focus of our continuing research within the Wireless IP project [21, 57]

[3] With current prediction performance [15, 56], a frame length of $3ms$ and a prediction range of $6ms$ would be more reasonable for mobile hosts moving at speeds up to $80km/h$

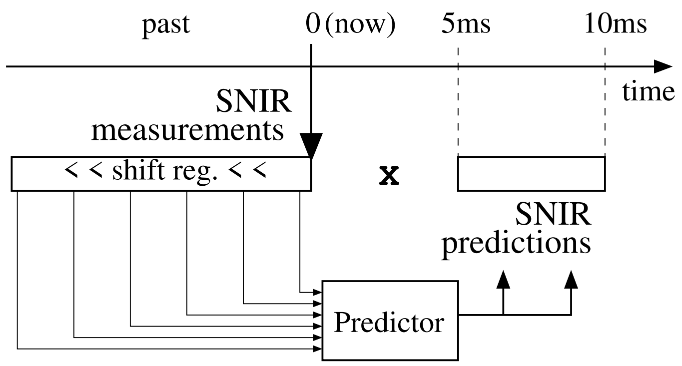

Figure 6.1: Channel SNIR measurements are continuously performed, and the estimations are fed into a buffer. The contents of the buffer is used by the channel predictor to obtain future estimations of the channel quality. These estimations (predictions) are collected for 5ms into one time frame. Between the completion of the prediction frame and the time marked with an "X", the scheduling has to be carried out. At the time marked with an "X", the scheduling decision has to be broadcast to the participating users, so that they get the remaining time to prepare for the reception of the next frame.

6.3 Inter-layer Signaling

The partition of responsibilities, described by the OSI-model (Appendix B), is natural and modular, but it also implies some stiffness in the communication protocols. For example, an interactive speech application over an IP network can tolerate some errors due to bad channel conditions, but the link layer protocol, that treats the stream as *data*, will do its best to reconstruct exactly the IP packet it received for transmission. There should be a way to tell the link layer that the data it is transmitting should preferably be transmitted *quickly*, rather than *correctly* [8]. Also, there should be a way for the link layer to tell the higher layers that the link provides some optional services, such as different levels of reliability in the transmission of packets. These possibilities are referred to as *inter-layer signaling*, and the scheduling approach results in an inter-layer signaling solution.

6.3.1 Realizing Inter-layer Signaling

From the link layer, where we have access to channel state information, the current service level could be announced to allow the higher layers to adapt to the constraint imposed on the transmission. Also, the applications currently running on the hosts should be able to require certain service levels from the mobile network and the wireless links.

This section outlines how this could be implemented in a real system.

The Buffering Subsystem

Apart from the subsystem that makes sure that we have channel predictions for all the ongoing sessions, we need an input buffer, the purpose of which is two-fold: First, we need some mechanism to arrange incoming packets from the wired network in the right order, so that this task is removed from the mobile host. Also, the packets are transferred into separate queues for different flows, so that a per-flow service can be offered by the link layer. At the same time, the buffer works as a shock-absorber between the two parts of the network, so that performance variations on either side, are partly hidden from each other. Second, we can use the buffer to estimate the amount of data that has to be transmitted to the different users. We also get the opportunity to analyze the contents of the data flows, so that different "economic values" can be assigned to the different flows. This information extraction is a part of the inter-layer signaling pipe, and provides the scheduler with necessary higher-layer information.

The buffer controller is assumed to submit a status report to the scheduling subsystem, described in the following section, so that the scheduler can make an appropriate decision on which queues to choose for the next transmission frame.

The Scheduling Subsystem

In the scheduling subsystem, the information from the higher and lower layers meet, and hopefully, the requirements and the service levels can be matched.

The contents of the scheduling subsystem is the subject for the remainder of this chapter, but first we will briefly discuss the FEC/ARQ subsystem.

The FEC/ARQ Subsystem

Even though the intention is that the scheduling subsystem shall make decisions and select appropriate coding and modulation complexities, to avoid re-transmissions, the possibility of an erroneous reception cannot be excluded. In the case of an erroneous reception, a NAK is sent back to the base-station, telling it to take appropriate measures for the event, depending on the application.

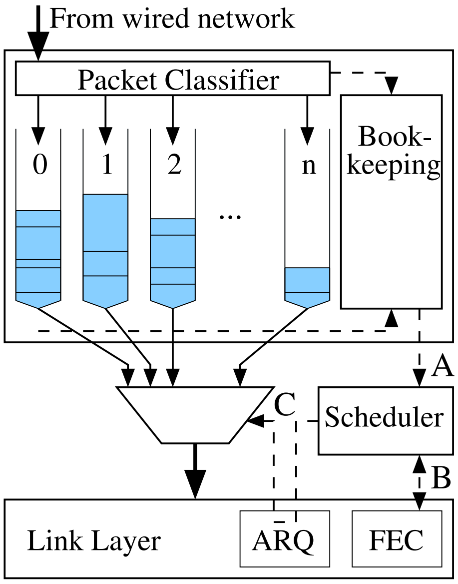

Figure 6.2: Schematic view of the buffer and its queues, and how they interconnect to the scheduler and the link layer. The packets arrive at the top and are inserted into their respective queues, restoring order among occasionally arriving out-of-order packets. The buffer regularly submits a status report (A) to the scheduler, containing information about the priorities, the size of the queues, and the required link service, some of which is also passed to the link layer (B). The scheduling decision (C) is updated by the link layer ARQ, and is then used to drain the queues.

In our approach, a Hybrid ARQ link layer protocol is assumed, as introduced in Section 5.6.1. More specifically, it is a Hybrid type-II ARQ system, using an outer punctured convolutional code, and an inner CRC code. The reason for using a hybrid ARQ system is that it provides incremental redundancy, that can be used as a fall-back in the case that the channel prediction fails. An ordinary ARQ system would not provide the combination of fine-grained adaptive coding, and ARQ.

Different services can be given to the network layer. One of the options is the number of attempts the link layer should try to transmit the data to obtain an error-free reception. (Completely error-free transmission, which would require an unlimited number of re-transmissions, is not provided by the link layer.) Table 6.1 shows three categories, named VOICE, MEDIA, and DATA, and some parameters for the service requirements they may have. For real-time applications such as telephony, where UDP might be the transport protocol, say, only three attempts should be made (Trans = 3 in Table 6.1) before giving up, and discarding the frame. For applications where TCP is used, the only constraint is the TCP re-transmission time-out that, when it is reached, will invoke a new segment to be sent from the source. In this case, up to eight transmission attempts (Trans = 8) could be made before giving up.

Table 6.1: Example of service requirement parameters for different types of applications. "Prio" stands for the priority of the packet flow. A higher priority value means that the flow should be given a higher priority in the resource assignment. "BER" is the desired target error rate for the transmission, and "Trans" is the number of link layer transmission attempts for each link layer word.

Class	Service parameters
VOICE	Prio=6 BER=10^{-3} Trans=3
DATA	Prio=2 BER=10^{-5} Trans=8
MEDIA	Prio=4 BER=10^{-4} Trans=3

The FEC and ARQ functionalities can be combined into one entity if Hybrid ARQ is used over the link, as discussed in Section 5.6, and in [22].

6.4 Optimization of Resource Allocation

Resource allocation deals with the problem of assigning shared resources to different tasks. In the case of wireless communication, the tasks comprise of the transmission of user data, and the shared resource is the radio spectrum, a resource, the quality of which varies with time and space. The available spectrum can be represented in many dimensions, such as time, space, frequency, code, and maybe even polarization, or combinations thereof. For spatial division we usually have the static cellular or sector approach, but this can also be dynamically assigned resources, by, for example, fast switching between neighboring base-stations when conditions allow and capacity profits from it.

The problem discussed and partially solved in this chapter deals with methods for (sub)optimally allocating time-slots to users. The allocation is based on the users' requirements and their predicted wireless channel conditions. The channel conditions are translated, via the target bit error rate (BER), to an allowed modulation format, see Figure 5.3, and Table 7.2. This translation results in an array of size $U \times S$, where U is the number of active users, and S is the number of time-slots in the scheduling window. This matrix will be called the *constraint matrix*, C. See Figure 6.3 where an example with six users and ten time-slots is outlined. Each entry in the matrix represents the number of bits per symbol

$$R = log_2 M \tag{6.1}$$

for the appropriate modulation format, with M-ary constellation. The modulation is selected based on the predicted channel quality for each time-slot and user, to give the highest transmission rate compatible with the target bit error rate for the user.

The required throughput is calculated from the amount of data currently in the input buffer, normalized by the time-slot size used in the radio link layer. If a user has a throughput requirement of 12, this means that she would be satisfied by 4 time-slots with uncoded 8-PSK modulation ($R = 3$).

The output from the scheduler is a vector with one entry for each time-slot ($1 \times S$), where each entry is the user number for the user that is selected to transmit in a particular time-slot. This vector is hereafter referred to as the *decision vector*, b. The decision vector can also be translated into a binary matrix of the same dimensions as the constraint matrix ($U \times S$), having one "1" for each time-slot. The "1" is in the location of the user that was allocated to that time-slot, and the "0"s are in the other locations. This matrix is the *allocation matrix*, X. The decision vector is then given by

$$b = (1, 2, ..., U)\, X \tag{6.2}$$

So, the problem is to generate a good-enough decision vector, from the given constraint matrix, and the requirement vector.

The resulting allocation vector, a, represents the resulting throughput over the ten time-slots from the schedule for each user. It is given by

$$a = diag\left(CX^T\right) \tag{6.3}$$

The resulting difference vector

$$\delta = a - r \tag{6.4}$$

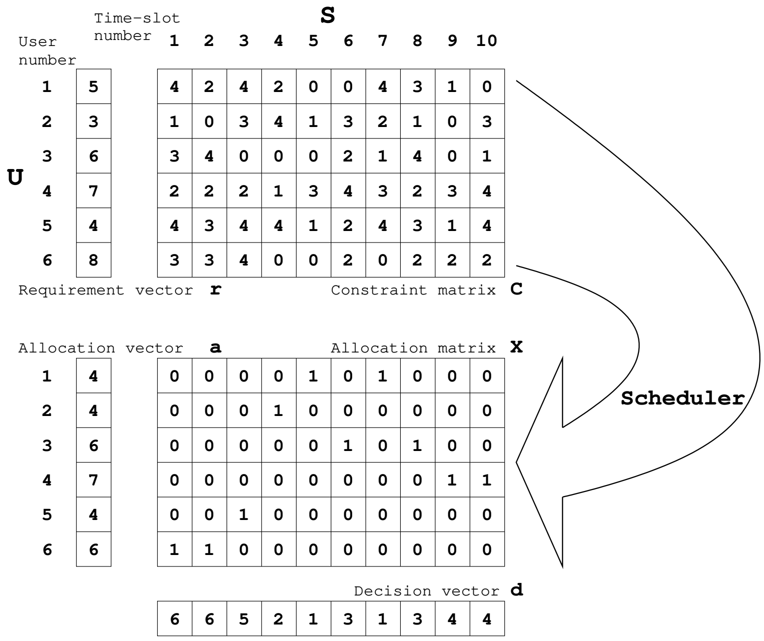

Figure 6.3: Example of a scheduling setup with six users and ten time-slots. The user requirement vector at the top left contains the desired resources for each user. The constraint matrix at the top right contains the allowed modulation formats for each user and time-slot. At the bottom is the decision vector indexing the user number that has been scheduled for a particular time-slot. It can be translated into the binary allocation matrix, and, through the constraint matrix (and Equation 6.3), the resulting throughput for each user is found in the allocation vector, at the bottom left. The schedule was calculated with the CSD algorithm, described in Section 6.5.4, and it can be seen that the allocation is not optimal. For example, we can see that if users 1 and 4 swap their slots 5 and 9, then user 1 gets the required throughput, which is not the case in the current schedule. Also, users 2 and 5 could swap their their slots 3 and 4.

represents unfulfilled requirements or unused resources. In this example, not all users' requirements were satisfied: User 1 and 2 would need to make some exchange, in order to meet the requirements. Also, user number 6 was under-supplied. This particular schedule was found by the CSD algorithm (see Section 6.5.4), and the drawback is that it cannot find the optimal solution in this case. In order to improve the schedule, the CSD algorithm would need to make more than one step, to find a better solution than the current one, and the first step has to be taken in a non-descending direction.

The set of admissible solutions to, and constraints on the scheduling problem constitute discrete values in a finite space. All solutions could therefore be found and classified by an exhaustive search, and the best one chosen. However, an exhaustive search becomes very complex. These kind of problems are referred to in the literature as NP-hard problems [4], meaning that the time it takes to find the solution is a Non-deterministic-Polynomial function of the dimension of the problem, also implying the impossibility of solving them in a general manner [32]. We will instead use numerical techniques that minimize a cost function.

6.4.1 Cost Functions

User Satisfaction

The cost functions we use in our optimization should somehow reflect our goal with the scheduling. An important goal is user satisfaction. (In this study, such quantities as *revenue maximization* are only dealt with indirectly, since they would require some pricing policy, and quantization of future goodwill, etc.) If all users are satisfied with the received service, we could assume that they happily will pay for it. One way of quantifying the instantaneous user satisfaction is by evaluating the difference vector $\delta = a - r$ between the resulting user throughput a, and the requirement vector r, from a scheduling decision. A negative element on δ would indicate that this user did not obtain enough bandwidth to transmit all his data. A positive value means that some of the channel bandwidth could be wasted by letting time-slots travel without data, unless new data has arrived into the buffers before transmission takes place. So, an optimal allocation in this sense would result in a difference of zero.

User satisfaction is however not only a function of throughput, but also of delay. Delay requirements can be incorporated into the cost function by introduction of priorities. Low-delay applications are associated with a high priority, and latency-insensitive data is given a low priority. Priorities may also change with time, reflecting the increasing urge of transmitting a pending real-time packet.

A cost function that includes all the considerations mentioned above, is the weighted squared norm of the difference vector, δ, expressed as

$$J = \|a - r\|^2_{P^\alpha} = \sum_u P_u^\alpha (a_u - r_u)^2 = \sum_u P_u^\alpha \left(\sum_s c_{us} x_{us} - r_u \right)^2 \tag{6.5}$$

where x_{us} is the (u, s) entry in the allocation matrix X for user u at time-slot s, r_u is the uth entry in the requirement vector r, that is, user u's required bandwidth according to the buffer status report (see Figure 6.2), and c_{us} is the (u, s) entry in the constraint matrix C. The weight P_u^α is a monotonically increasing function of the priorities for the different users, that might even be time-varying. The exponent $0 \leq \alpha \leq 1$ can be used to manipulate the behavior of the scheduler in different traffic scenarios. An α close to zero reduces the impact of the individual stream priorities, thus making the total throughput more important, while an α close to one takes into full consideration the individual priorities.

Throughput Maximization

A different way of regarding the optimization problem is from the point of view of system throughput. In the schedule optimization context, system throughput T is defined as the average number of bits per symbol transmitted in the S time-slots[4], c.f. Figure 6.3:

$$T = \frac{1}{S} tr \left(C X^T \right) = \frac{1}{S} \sum_{u=1}^{U} \sum_{s=1}^{S} c_{us} x_{us} = \frac{1}{S} \sum_{u=1}^{U} a_u \tag{6.6}$$

In a throughput maximization, we would like to maximize the system throughput, with the (somewhat "soft") constraints of also keeping as many users as possible satisfied with their received service. Thus, the user satisfaction comes in as a constraint instead of as a cost function. This viewpoint results in a more difficult problem to solve, suggesting linear programming solutions, since throughput is a linear function of the allocation matrix. The difficulty is in the inclusion of the constraints: The constraints are "soft" in the sense that they need not necessarily be fully met; all users do not necessarily have to be satisfied for the solution to be feasible. However, a linear program can only take "hard" constraints into account, leaving us with the option of starting with a solution to an unconstrained maximization problem and introduce the constraints sequentially, in a narrowing fashion. This approach has been avoided, due to the unattractiveness of the problem formulation.

6.5 Numerical Algorithms for Scheduling

We now focus on the user satisfaction criterion (6.5), and outline different possible algorithms for numerical optimization. It is desirable to employ

[4]Note that the real throughput is found by multiplying T with the number of symbols per time-slot.

a simple scheduling algorithm. It should however be powerful enough to improve the over-all performance of the communication system. A number of different methods for time-slot allocation have been implemented and compared. These are presented next.

6.5.1 Optimal Allocation by Exhaustive Search

This is not a viable solution to the scheduling problem, since the optimal allocation performs a search through all combinations of time-slot allocations. This algorithm has been implemented, but due to its high computational cost when applying it on a relevant example, it has not been used. When having U users competing for S time-slots, the algorithm has to try U^S schedules. For the example in Figure 6.3, with 10 time-slots and 6 users, the number of schedules is $6^{10} \approx 60.5 \times 10^6$.

6.5.2 Maximum Allocation

This time-slot allocation algorithm is extremely simple and fast. The decision is simply to give each time-slot to the user that has the highest predicted throughput in that time-slot. The drawback is obviously that it does not consider the amount of data that each user has waiting in its buffered queue. So the probability is quite high that a user might be over-supplied, that is, that he is allocated more resources than he can actually utilize. Consequently, some users are also probable to become under-supplied.

6.5.3 Best First (Non-predictive Scheduling)

This approach, which is also very simple, simply goes through the time-slots in chronological order, giving them one by one to the user that has the highest throughput in that time-slot, provided the user is under-supplied. In case this user is satisfied, the time-slot goes to the user with the second-best throughput, and so on.

This approach can also be regarded as a non-predictive scheduling, or link adaptation, only taking present conditions into account when assigning time-slots.

6.5.4 Controlled Steepest Descent

CSD utilizes the criterion (6.5) for evaluation of the instantaneous overall user satisfaction, and employs a steepest descent search among the feasible solutions to the scheduling problem. By re-allocating one time-slot at a time, and doing that for the transaction that results in the largest decrease in (6.5), we hope to find a near-optimal schedule:

1. First a good initial guess for the schedule is needed. Such a guess has been found to be the maximum allocation, described above.
2. Iterate for each time-slot (S time-slots)
   (a) Evaluate the reduction of (6.5), when an assigned time-slot is given to the other users ($U-1$ evaluations)
3. After all the possible one-step transactions have been evaluated, the one giving the highest increase in over-all user satisfaction (or equivalently, the biggest decrease in over-all user pain) is executed.
4. Now we have a slightly better (in the sense of (6.5)) schedule.
5. Use this as the initial guess and iterate the procedure, until no decrease in (6.5) is accomplished.

In each step, $(U-1) \times S$ calculations of (6.5) are done. A discussion on the feasibility of this algorithm is given in Appendix A.2. Moreover, as can be seen from the example in Figure 6.3, where the CSD algorithm was used, the optimal solution is not obtained. For example, we can see that if users 1 and 4 swap their slots 5 and 9, then user 1 gets the required throughput, which is not the case in the current schedule.

Also, users 2 and 5 could swap their their slots 3 and 4, which results in a lower scheduled system throughput, but also a lower value of the cost function (6.5), which is the objective. The failure to in finding the optimal solution is due to that the remaining steps to an optimal solution would require *two* simultaneous transactions (in step 2 above) in order to show an increase in user satisfaction.

6.5.5 Robin Hood

This is a simplification of the CSD (Controlled Steepest Descent) algorithm outlined in the previous section, but without evaluation of the quadratic cost function. The scheduler performs the scheduling in two rounds. The first round is simply the maximum allocation, described in Section 6.5.2, where each time-slot is allocated to the user that can transmit at the highest rate in that time slot (unconstrained maximization). In the second round, time-slots are redistributed from users that have been over-supplied (rich), to users that have been under-supplied (poor), with respect to their required throughput. We call this equalization to user satisfaction the Robin Hood principle: To take from the rich, and give to the poor. This algorithm is simple and it works as follows:

1. The initial allocation is done by the Maximum Allocation algorithm, described previously

2. Find the rich and poor users by comparing their allocations to their amount of data in the queues

3. Loop until either no more rich or no more poor users exist:

   (a) For the richest user, find its worst time-slot, in the sense of providing the lowest predicted transmission rate.

   (b) Among the poor users, find the best user (in terms of the predicted transmission rate) in that time-slot, and give the time-slot to him. In case two poor users have the same transmission rate, choose the one with the higher priority.

   (c) Update the rich and poor variables.

A crucial requirement for the algorithm to converge is that never must any rich users become poor, or vice versa. This is realized by having a gap between the rich and the poor domains, too big to be crossed by one redistribution step.

As can be seen from the algorithm, the priority for different flows is a criterion that is used only when two users have the same predicted throughput in a time-slot. When the throughput is the same for the two best users, the one with the higher priority is chosen. In the case that they have the same priority as well, their predicted channel quality is used for comparison. The three features *throughput*, *priority*, and *predicted SNR* can also be compared in different order, when looking for the best user to receive a time-slot (in step 3b above), resulting in different performance (see figures 7.6 and 7.7).

6.6 Channel Prediction Quality vs. Scheduling Performance

The quality of the predictions affect the final outcome, when the scheduled decisions are executed by transmitting the data for the different users at the allocated time-slots. It should be obvious that prediction is difficult, and that the difficulty increases with an increasing prediction range. Thus the accuracy should be expected to decrease, as prediction range increases, which was also seen in Figure 2.3. These results have been shown in [15].

The interesting part for the current study, is how the decreased accuracy influences on the scheduling and transmission performance. Simulations have been conducted in order to evaluate the different approaches to link-layer transmissions, with and without scheduling and prediction. These simulations and the results thereof are described in the following chapter.

Chapter 7

Simulations

In this chapter, simulation results will be presented along with the underlying assumptions, in terms of idealized conditions and simplifying abstractions. The goal is to evaluate the performance that is obtained through the use of channel predictions and scheduling of the available resources.

Section 7.1 only considers the scheduler's ability to meet the users' requirements with the provided channel resources. In Section 7.2 we take the simulations one step further, by evaluating how the scheduling decision affects the resulting throughput and user satisfaction at the link layer, assuming perfect predictions. Finally in Section 7.3, we introduce errors in the predictions, and evaluate the the performance as a function of a varying channel prediction accuracy.

7.1 Scheduling Algorithm Performance Comparison

In this section, the performances of three different scheduling algorithms are presented. Three measures are compared, namely, the *throughput* as defined by (6.6), the *user satisfaction*, as measured by (7.4), and the *iterations* required for convergence. These simulations consider only the performance of the scheduler itself. Some of the results have been published in [19].

7.1.1 Simulation Setup

The size of the scheduling problem in these simulations is $U = 9$ users, competing for $S = 48$ time-slots. The traffic amount is adjusted to be a little more than can be accommodated in the system, to make the scheduling problem interesting. The number of possible solutions to this problem is $9^{48} \approx 6.4 \cdot 10^{45}$, which clearly excludes the possibility for an exhaustive search, especially since the experiment is repeated for 1000 time-frames to obtain relevant statistics.

The constraint matrix C is generated from a discrete (integer) uniform distribution between 0 and the maximum allowed modulation format (bits/symbol). The requirement vector that reflects the amount of data in the buffers for the different users, r, is generated from a normal distribution, according to

$$r_u = \lfloor \frac{S\,(R_{max} + z_u)}{U} \rfloor \qquad u = 1, ...U \tag{7.1}$$

where z_u is drawn from a zero-mean, unit variance, normal distribution. S is the number of time-slots in each schedule, U is the number of participating users, and R_{max} is the maximum allowed modulation level in bits per symbol. The delimiters $\lfloor \cdot \rfloor$ denote a rounding operation downwards to the nearest integer (a negative value is replaced by 0). For each new simulated schedule, a new requirement vector is created, without taking the requirements remaining from previous schedules into the calculation. By doing this, we only investigate the instantaneous performance, and not the long-term behavior of the queue lengths due to the scheduling strategy.

The average throughput (assuming that no clever resource allocation taking place) that would be allowed by the simulated system (represented by the constraint matrix C) is

$$S \times \frac{1}{5} \sum_{c=0}^{R_{max}} c = 48 \times \frac{1}{5} \sum_{c=0}^{4} c = 96 \tag{7.2}$$

and the offered traffic has an average of

$$U \times \lfloor \frac{S\,(R_{max} + z_u)}{U} \rfloor = 9 \times \lfloor \frac{48\,(4+0)}{9} \rfloor = 189 \tag{7.3}$$

so the system is slightly overloaded.

The user satisfaction has in Section 6.4.1 been defined as the difference between the allocated and the required resources for one user u within one scheduling frame, that is, the components of the difference vector δ:

$$\delta_u \equiv a_u - r_u = \sum_{s=1}^{S} c_{us} x_{us} - r_u \tag{7.4}$$

A negative value on the user satisfaction means that the user was not given enough resources to meet its requirements.

User satisfaction is also measured in terms of *unfairness* of the scheduler output. The unfairness, Δ, is defined as the difference in user satisfaction, between the best and the worst satisfied users within one scheduling frame:

$$\Delta \equiv \max_u \{\delta_u\}_{u=1}^U - \min_u \{\delta_u\}_{u=1}^U \tag{7.5}$$

Unfairness as defined by (7.5) has little in common with traditional fairness measures, that reflect the proportions of the bandwidth allocated to different flows, regardless of variations in the resource availability [39].

7.1.2 Simulation Results

The results are presented in Figures 7.1 through 7.4 by means of histograms, displaying the distribution of the simulated scheduling outputs. In Figure 7.1 we see the throughput resulting from (top to bottom) Best First, Robin Hood, and CSD, and in Figure 7.2 the resulting unfairness. The user satisfaction is displayed in Figure 7.3, and in Figure 7.4 the number of iterations for each method is presented.

In Figure 7.1 we can see that the resulting throughput from the three scheduling methods Best First, Robin Hood, and CSD, respectively. As previously stated, throughput in the scheduling context is defined in (6.6), which is a measure of how how well the resources are utilized by the system. Note that the maximum system throughput is 4 bits per symbol in this case. On average, we manage to use $3,5$-$3,7$ of these bits over each time-slot. In these plots, the numbers on the x-axis represent the average rate at which the scheduled transmission would have been performed, in bits per symbol.

We can see that Best First results in a somewhat higher average throughput than Robin Hood, and that the throughput from Robin Hood is slightly higher than for CSD. It is tempting to state that the much simpler algorithm Best First, also is better, but a glance at Figure 7.2 tells us otherwise.

In Figure 7.2 the resulting unfairness, as defined in (7.5), is depicted for the same schedules as in Figure 7.1. The unfairness reflects the difference in user satisfaction within one schedule, that is, the difference between the most satisfied and the least satisfied users. Here we can see that Best First is extremely unfair, whereas Robin Hood is less unfair, and CSD is the least unfair of the three algorithms. This suggests that there has to be a trade-off between unfairness and throughput. A method that achieves a lower average unfairness, such as CSD, has to pay with a lower resulting throughput.

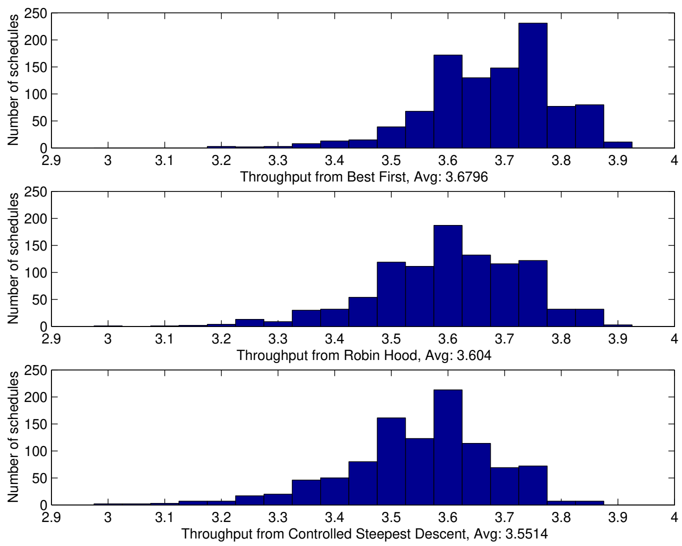

Figure 7.1: Resulting system throughput for different optimization approaches. The top histogram shows the result from the Best First algorithm (unpredictive scheduling), the middle histogram shows the values for the Robin Hood algorithm, and finally, the bottom histogram shows the results for the CSD algorithm. The y-axis gives the number of schedules resulting in the throughput on the x-axis. The system throughput on the x-axis is represented by the average scheduled transmission rate in bits per symbol in each schedule. The more schedules on a higher x-value, the better throughput. We see a slightly decreasing throughput for increasingly complex scheduling algorithms.

From Figure 7.3, depicting the user satisfaction from the same schedules as above, it is also evident that the different search methods result in different solutions. In the Best First algorithm we see that we have a high weight of satisfied, and even over-supplied, users (the big bump in $0 \leq \delta_u \leq 4$), but also that we have a long tail in $\delta_u < 0$, consisting of very unsatisfied users. For the Robin Hood method, a tail also exists on the positive side (very over-provisioned users), whereas with the CSD algorithm, tails are completely removed. Removing the tails comes at the cost of accepting a slightly lower average user satisfaction. The fact that the user satisfaction on average is negative for all cases is expected, since the simulated system is slightly overloaded.

The number of iterations is summarized in the diagrams of Figure 7.4. These results exclude the initial guess, which is achieved through the Max-

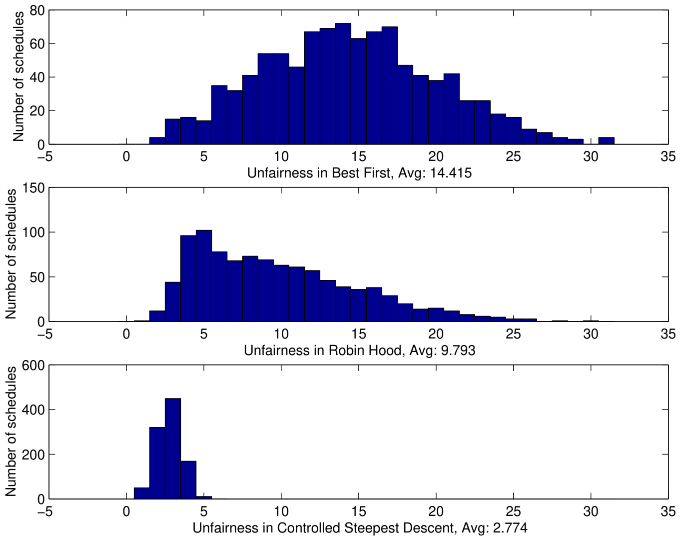

Figure 7.2: Unfairness, as defined by (7.5), among users for the different scheduling algorithms. The top histogram shows the result from the Best First algorithm, the middle histogram shows the values for the Robin Hood algorithm, and finally, the bottom histogram shows the results for the CSD algorithm. The y-axis shows the number of schedules resulting in the unfairness on the x-axis. The more schedules on a lower x-value, the better. We see that an increasingly complex scheduling algorithm results in a decreasingly unfair, thus better, schedule. We also see the difference in shape of the unfairness distributions, reflecting the difference in ability to reach a global minimum in the schedule optimization. The CSD algorithm (bottom diagram) results in a narrow distribution since it is capable of finding a near-global minimum in most cases, whereas Best First (top diagram) more seldom does that.

imum Allocation, described in Section 6.5.2, and is the same for all the described scheduling algorithms.

Note that the computational complexity per iteration is much higher for the CSD algorithm than for the other two algorithms. For each step in the Controlled Steepest Descent algorithm, 8×48 sums of 48 squares are calculated, whereas in the other two algorithms, merely a maximum value in a 9-element vector is found for each iteration. So, in comparison, this version of the CSD is extremely complex[1], but it is also extremely fair. Still, the computational complexity of CSD is quite reasonable. See Appendix A.2

[1] It is quite straightforward to reduce the complexity so that only four squares are performed instead of 48, but at the cost of higher storage requirement.

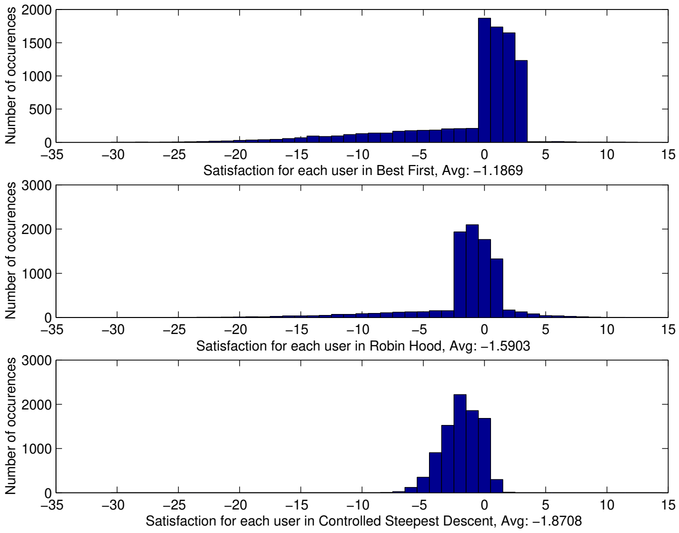

Figure 7.3: User satisfaction in terms of the difference between the allocated and the required resources. The resource is represented on the x-axis as the number of time slots with BPSK modulation (δ_u). A negative value on δ_u means that the user was under-supplied by a schedule. There were 1000 schedules with 9 users in each, so the total histogram weight is 9000. The long tails in the top two graphs reflect the inability of finding an efficient allocation for all users within one schedule.

for a brief feasibility study of the CSD method.

In Figure 7.4, a positive feature of the Robin Hood algorithm becomes apparent, and that is the limited and small number of iterations for convergence to a solution. Since we have $S = 48$ time-slots, at most 48 re-distributions will take place before the algorithm returns its decision. This is thanks to the concept of rich and poor users, and to the constraint that no rich user is allowed to become poor, and vice versa, as described in Section 6.5.5.

An interesting observation (Figure 7.5) from simulations of the CSD algorithm, is that the initial guess is very important to prevent the algorithm from converging to a local minimum. The diagrams to the left show the amount of unfairness in the resulting schedules, and the diagrams at the right show the number of iterations taken by the CSD algorithm for convergence. The preceding Maximum Allocation works well with the CSD algorithm (two top histograms), but if we instead run the schedule through the Robin Hood algorithm before the CSD algorithm (two bottom histograms), then the initial schedule comes near a local minimum (left histogram), to

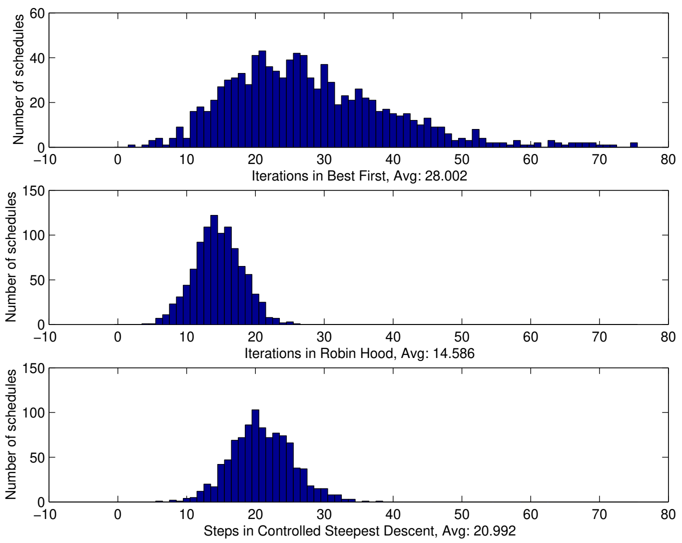

Figure 7.4: Required number of iterations for the different scheduling algorithms to converge. The two top diagrams have equivalent computational complexities, which is merely a selection of the largest element in an array of size $U = 9$ users. The CSD algorithm, in the last diagram, however, has a higher complexity in each iteration (see Appendix A.2 for a brief discussion of the required calculation effort). The histograms show the number of schedules as a function of the number of iterations. The more schedules on a lower iteration value, the better. Note that the Robin Hood algorithm has the ability of converging in at most 48 iterations, which is the number of time-slots (S) in a scheduling frame, so no schedules have a higher number of iterations than 48 for the Robin Hood algorithm.

which the CSD algorithm converges in very few steps (right histogram). This is seen in that the resulting unfairness distribution is similar to the one from the Robin Hood algorithm (in Figure 7.2), and in that the number of steps taken by the CSD algorithm (right histogram) is very small. The behavior is similar when we precede the CSD algorithm with the Best First algorithm (two center histograms), but the high peak at the left of the left-hand histogram indicates that, in many cases, the CSD algorithm manages to converge almost to the global minimum. The remaining bump centered at about $\Delta = 15$ in the left-hand histogram, indicates that many schedules also remain at their Best First solution.

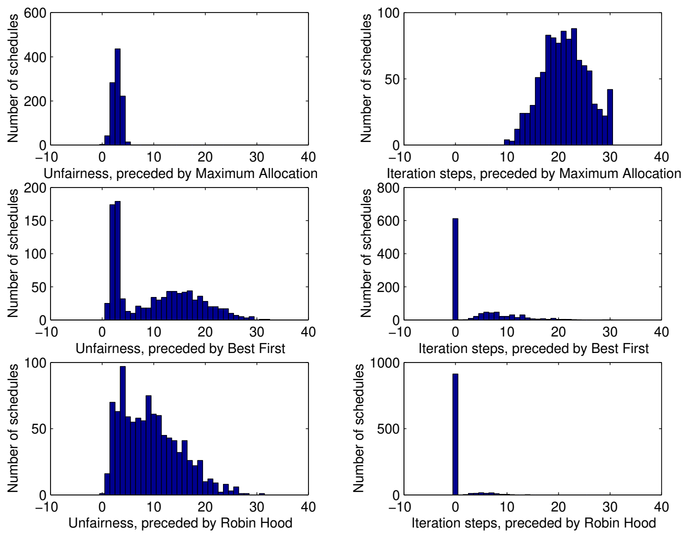

Figure 7.5: Comparison of unfairness (left) and number of steps for convergence (right), when the Controlled Steepest Descent (CSD) algorithm is preceded by other scheduling algorithms. It is evident that the Maximum Allocation (top) is a good candidate for an initial solution for the CSD algorithm, whereas Best First (middle) and Robin Hood (bottom) do not provide a good start for the CSD algorithm to find a global minimum. They apparently converge to a local minimum, that the CSD algorithm cannot escape from.

7.1.3 Conclusions

The Controlled Steepest Descent (CSD) algorithm shows attractive performance in terms of unfairness. Its complexity is not unreasonable even in a naïve implementation, see Appendix A.2, and it is simple to further reduce its computational complexity by a factor 10 in the present example.

The Robin Hood algorithm, is fast and not as embarrassingly unfair as the Best First algorithm. The Best First algorithm does not even use the knowledge from the long-term predictions, and would perform equally (bad) with instantaneous channel estimates.

For the continuation of the thesis, the Robin Hood algorithm will be used in the simulations. The reason for this choice is that the Robin Hood algorithm shows

- Low complexity
- Reasonable fairness

- Fast convergence

7.2 Transmission Performance

In this section, the simulations are taken one step further, carrying out short runs of link layer transmissions using the Robin Hood scheduling approach. Robin Hood was chosen for its attractive features that motivate a thorough investigation. These results are based upon previous work, partly described in [17, 18, 19, 23].

7.2.1 Simulation Setup

The simulated system consists of many different objects that interact, to create a realistic setup for simulated evaluation. First, traffic sources generate packet based traffic according to different models. The generated traffic is then received at a simulated gateway node with packet classification and per-flow buffering abilities. The status of each user's buffer is reported to a resource scheduler, including information of destination, traffic class, and queue length (requirement vector). The channel state for each user for the next $5ms$ frame is reported to the scheduler. Based on the channel state and the traffic class, the possible transmission rates are calculated for each user and time-slot, resulting in the previously described constraint matrix C (see Section 6.4). For the simulations in this section, however, the maximum allowed transmission rate is 6 bits per symbol (64-QAM). The scheduling is carried out with the Robin Hood algorithm, and the schedule is reported to the link layer. The link layer adjusts the schedule to fit possible re-transmissions into the allocated resources, and then drains the queues to fill the remaining allocated resources. The data is channel encoded, then transmitted over a simulated Additive White Gaussian Noise (AWGN) channel, and finally, the data is decoded and possible re-transmissions are requested with the NAK feedback[2].

In this idealized scenario, we assume that the channel conditions are predicted without error for 10 milliseconds ahead in time. Moreover, perfect synchronization and transmission at a constant maximum amplitude, regardless of the symbol alphabet, is assumed. The experiment is applicable to both TDD and FDD systems, provided that accurate predictions of the channel conditions exist.

[2]No Forward Error Correcting codes (FEC), nor Automatic Repeat reQuest (ARQ), are implemented in the simulations described in Section 7.2.2.

Traffic Sources

In the simulations we use three traffic types: Voice, Data, and Media, see Table 7.1. The incoming traffic was generated using a Poisson distribution for the packet inter-arrival time [32, 64], and a Pareto distributed packet size [13], except for the "VOICE" traffic class, which was chosen to have a fixed packet size to comply with current wireless standards. The Poisson cumulative distribution function is given by (7.6), and the Pareto cumulative distribution by (7.7).

$$F(t) = 1 - e^{-\lambda t} \tag{7.6}$$

$$F(t) = 1 - \left(\frac{k}{t}\right)^{\alpha}, \quad t \geq k \tag{7.7}$$

The parameters are λ, which is the inverse of the mean inter-arrival time, k which is the minimum packet size in the Pareto distribution, while α, is a shape parameter[12, 13]. For $\alpha \leq 1$ the Pareto distribution has an infinite mean, and for $\alpha \leq 2$ it has an infinite variance. The different traffic classes and the parameter choices used in the simulations are presented in Table 7.1

	Service parameters		Source parameters	
Class	P(err)	Prio	Inter-arrival	Packet size
VOICE	10^{-3}	6	$\lambda = 1/0.001$	Fixed, 256
MEDIA	10^{-4}	4	$\lambda = 1/0.005$	$k = 1280, \alpha = 1.03$
DATA	10^{-5}	2	$\lambda = 1/0.01$	$k = 8000, \alpha = 1.03$

Table 7.1: The different traffic classes used in the simulations and how their parameter values are chosen. A high value on k means large packets. Larger λ generates packets more often. A higher value on the priority means higher priority. The fixed packet size of 256 for the VOICE class, means that a VOICE packet has the size of 256 bits. Moreover, $k = 1280$ means that the minimum packet size in the Pareto distribution is 1280 bits.

Traffic Classes

The traffic classes presented in Table 7.1, have traffic source parameters with related service parameters. The target error rates, P(err), can be approximately found from, for example, Figure 5.3, where the bit error probabilities have been approximated with the symbol error probabilities. See Appendix A.3 for details on how to find the modulation decision thresholds for different target symbol error rates and modulation alphabets, in the case that no channel coding is used. In a "real" implementation of a scheduler of this type, the thresholds should be selected based on simulations, or by performance measurements on real traffic.

Buffers and Queues

Each generated packet is received by the buffer, then scanned for source and destination, and then inserted in the right queue (see Figure 6.2). Each initiated queue is associated with a priority and a source-destination pair. The buffer state is updated by increasing the queue lengths with the sizes of the newly arrived packets. Similarly, when the link layer drains the queues, the buffer state is updated by decreasing the respective queue lengths. Network-layer packet boundaries do not affect the amount of data drained from the queues in each link-layer word.

Channel State Predictions

To simulate channel state predictions, the channel measurements described in Section 6.2.1 where used by selecting 48 channel samples for each scheduling frame, one sample for each time-slot. The predictions were assumed to be perfect, so the same values are used for the actual channel simulations, described next. In the simulations described in Section 7.3, however, the predictions are not assumed to be perfect.

AWGN Channel Simulations

The data bit stream is modulated and transmitted with a constant maximum amplitude over the noisy channel. White Gaussian noise with varying variance is added to simulate changing channel conditions. Thus, we use a one-tap fading channel, where the fading is simulated by varying the noise variance. The SNR values used for the channel simulations are taken from the same channel measurements (Section 6.2.1) as for the channel state prediction simulations, described above. At the receiver, the signal is demodulated and the obtained bit stream is compared to the original one. The number of errors is counted, as well as the number of transmitted bits.

7.2.2 Transmission Simulation Results

In Figures 7.6 and 7.7 the outcome of two simulations are presented. In Figure 7.6, the top graphs show the bit throughput represented by a vertical bar for each frame, and a different color/nuance of gray for each of the

25 users. The bottom diagrams show the resulting bit error rate (BER). Each frame consists of 48 time slots. Each time slot corresponds to one output sample from the channel predictor. The scheduler is optimizing the transmission within each frame.

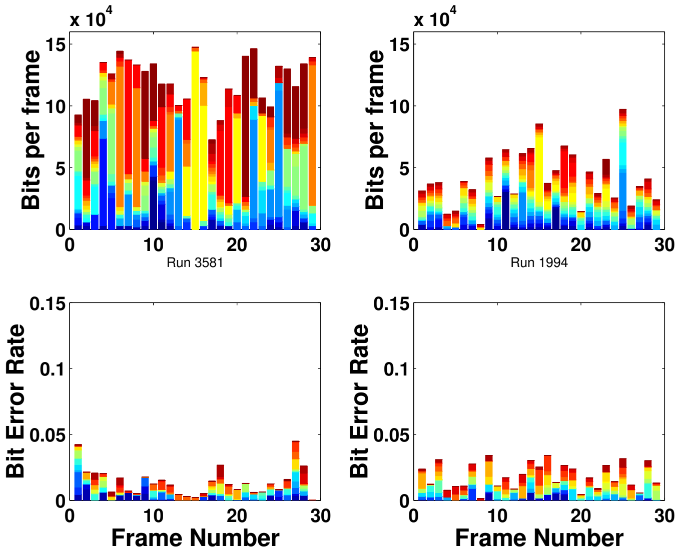

Figure 7.6: Scheduling and transmission result after running the Robin Hood scheduler on two equal sets of 25 users and their respective channels, but with different settings on the order of comparison *of the parameters in the scheduling process. To the left,* modulation format *was compared before* user priority. *To the right,* user priority *was compared before* modulation format. *This is reflected in the resulting lower throughput at the right.*

The scheduling algorithm was modified between the two simulations, by interchanging the parameters used for classification of the different users and channels. To the left, *modulation format* is compared before *user priority*, and, to the right, vice versa. As indicated by Figure 7.6, the parameters used in the scheduling process have a large impact on the performance of the scheduler. Note that the total channel capacity is utilized if the whole square in the top figures are filled.

The delay performance of the scheduler decisions is depicted in Figure 7.7. The delay is measured from the arrival of the packet, at the incoming buffer, to the end of the time-frame in which the packet was completely transmitted. The absolute values of the delays should not be given too much importance, since they depend on the packet size, the packet inter-arrival time, the time-

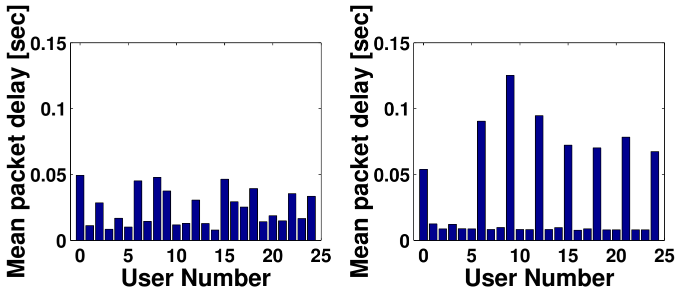

Figure 7.7: The delay profile for the 25 different users included in the simulation depicted in Figure 7.6, averaged over ten simulations. To the left, the average delay is lower, since the modulation format *is compared before the* user priority*. To the right, we see a more differentiated delay profile, since higher priority users always get access to the channel before lower priority users. The users belong to the classes MEDIA, VOICE, VOICE, ..., MEDIA, VOICE, VOICE, MEDIA from left to right.*

frame size, and the desired error probability. One should instead compare the two diagrams in Figure 7.7, and conclude that the scheduler performance can be adapted to the traffic conditions.

7.3 Varying Prediction Accuracy Simulations

In this section we carry out a large number of simulations, basically with the same setup as in the previous section, but now we remove the assumption of perfect channel predictions

The channel state information given to the scheduler is altered by addition of a normally distributed term, to the true state, to simulate inaccurate predictions that deviate from the values used for the actual channel simulation. Another difference from the previous simulations, is that the traffic sources are "boosted", so that they generate so much traffic that the probability of a queue being empty, is negligible. In practice, this is achieved by setting the packet sources in Table 7.1 so that they generate 8kbit packets at a higher rate ($\lambda = 1/0.001$). Different levels of link layer data protection, such as Hybrid type-II ARQ, are also introduced. Moreover, the modulation alphabets used are presented in Table 7.2 for the case of adaptive modulation without ARQ. For the simulations including Hybrid type-II ARQ the corresponding limits are more extensive, since they not only involve the different modulations used, but also the different code rates for each re-transmission.

The reader is referred to [19] for a description of the thresholds in this case. The purpose of these simulations is to find *how well we need to predict the channel quality in order to make the scheduling approach work*. Also, the question can be reversed: We want to find *how long predictions we can use in order to have an efficient transmission at the link layer*, given a certain prediction accuracy.

The simulations are carried out for 29 scheduling frames, with 48 time-slots in each frame. There are five users in each of the three traffic classes, so the scheduling problem is solved in each frame by allocating 48 time-slots among these 15 users with different requirements.

7.3.1 Robin Hood with Adaptive Modulation

In the first set of simulations, the predictions are only used for the selection of an appropriate modulation alphabet, to fulfill a pre-specified target bit error rate according to Table 7.2. The only redundancy that is added is a checksum (CRC), to determine whether the received link layer word[3] was correct or not. There is no mechanism to handle re-transmissions at the link layer, so an error in a received word could be handled in two ways: Either the word is dropped, probably leading to re-transmissions commanded by higher layer protocols. Or, the word could be kept, then merged with other link layer words into a network layer packet, and then passed to the higher layers at the receiver. If the higher layers accept some errors in the payload data, then there is a probability that the packet is useful in spite of some errors [8].

The thresholds used for different modulations have been found from BER link simulations, as outlined in [19]. For these simulations, only four different modulations were used, namely BPSK, QPSK, 8PSK, and 16QAM. The thresholds for the uncoded case are given in Table 7.2, and the target error rates are presented in Table 7.1.

The performance is evaluated for different accuracies in the prediction of channel quality, and the results for the uncoded case are shown in Figure 7.8. For a TCP segment of size 1000 bits, approximately 5 link layer words have to be correctly received in order to provide a correct TCP segment. A word error rate of about $10^{-2} = 1/100$ would result in 1/20 of the TCP segments to be in error. If ordinary TCP without selective acknowledgments (see Section 4.1.2) is used, then the performance would become unacceptable due to the high number of transport layer re-transmissions [27, 37]. For the

[3]A link layer word consists of 216 bits, including 12 CRC bits.

BER	BPSK	QPSK	8PSK	16QAM
10^{-1}	-0.86	1.94	5.64	7.91
10^{-2}	4.32	7.33	12.1	13.9
10^{-3}	6.79	9.80	14.8	16.6
10^{-4}	8.40	11.4	16.5	18.2
10^{-5}	9.59	12.6	17.7	19.5

Table 7.2: The thresholds in channel SNR , measured in dB, for different modulation formats to achieve the target bit error rates (BER) at the left.

uncoded case, we then conclude that a prediction error standard deviation of around $2dB$ and more (bottom left in Figure 7.8) would result in an unacceptable performance at the transport layer for the DATA traffic class. On the other hand, if the transport protocol is UDP with a limited header checksum, then the error rate may be acceptable for some applications, such as the VOICE traffic class. It is also probable that a simple block code with error correcting capabilities can help improve error performance, and this is a topic for further research.

7.3.2 Robin Hood with Hybrid type-II ARQ and Adaptive Modulation

This second set of link layer simulations include an advanced ARQ mechanism, described in Section 5.6.1, that enables discovery and re-transmission of erroneous link layer words.

These simulations show that we can obtain a high throughput and still maintain an error free transmission of data, when backing up the (inaccurate) scheduler with the ARQ mechanism, see Figure 7.9. A prediction error standard deviation of up to $3,5dB$ can still be used for scheduling purposes, when there is an auxiliary re-transmission mechanism, such as the Hybrid type-II ARQ, for DATA[4] and MEDIA, but at the cost of reduced throughput and increased re-transmission delays. For a prediction error standard deviation of up to $2dB$, the throughput and delay performances are not significantly affected by the increased prediction error standard deviation, as

[4] The DATA traffic class shows some unexpected behavior that we have not been able to explain. For a prediction error standard deviation above $3,5dB$, the word error rate decreases and the throughput increases. One reason could be insufficient statistics in the simulations, due to the limited amount of available channel measurements. Another reason could be that the scheduler allocates the resources to the other two traffic classes, decreasing the statistics even more for the DATA traffic class.

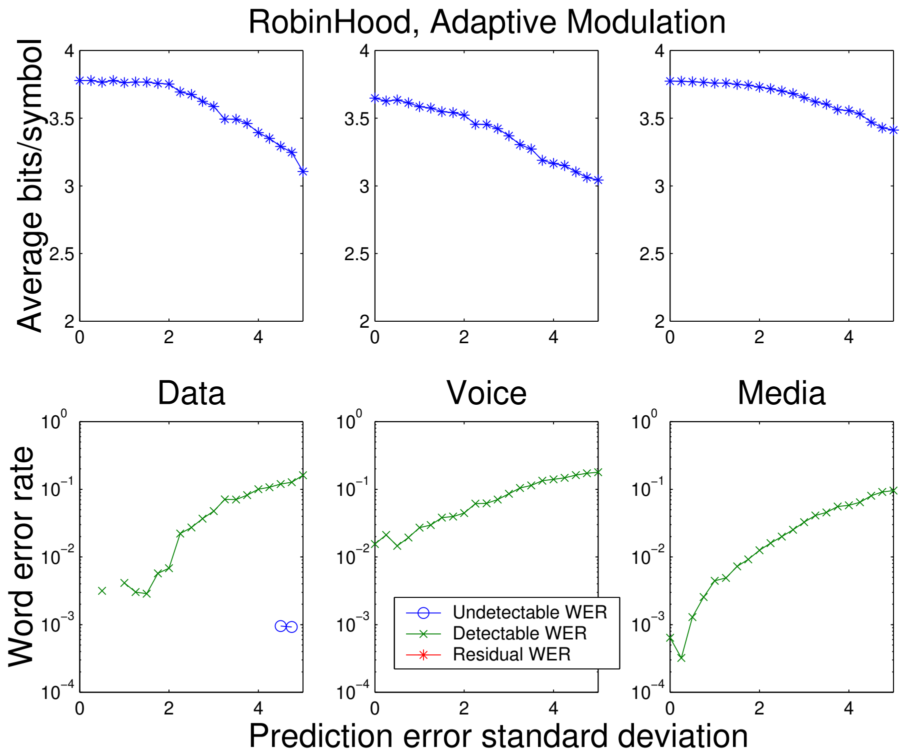

Figure 7.8: Throughput (top) and resulting word error rates (bottom) when transmitting at a rate decided from prediction of channel quality. The prediction error standard deviation is varied along the x-axis of the plots, with no error to the left (0dB), and a standard deviation of 5 dB around the actual value, to the right. The detectable error rate is the same as the residual error rate, since there are no re-transmissions, and the detected erroneous words are simply dropped. Undetected errors (bottom, left) result in an erroneous word being assembled with other words, to recreate a network layer packet, which will be in error.

compared to a perfect prediction.

The residual error rates in the VOICE traffic class may not be of high importance, since the transport protocol for such a service would be UDP, and the UDP checksum can be chosen not to cover the payload. It is, however, interesting to note that the VOICE throughput does not decrease much from the non-coded case (Figure 7.8) to the Hybrid type-II ARQ case in Figure 7.9, for a prediction error standard deviation up to $3,5dB$. The residual error rate is, however, considerably reduced.

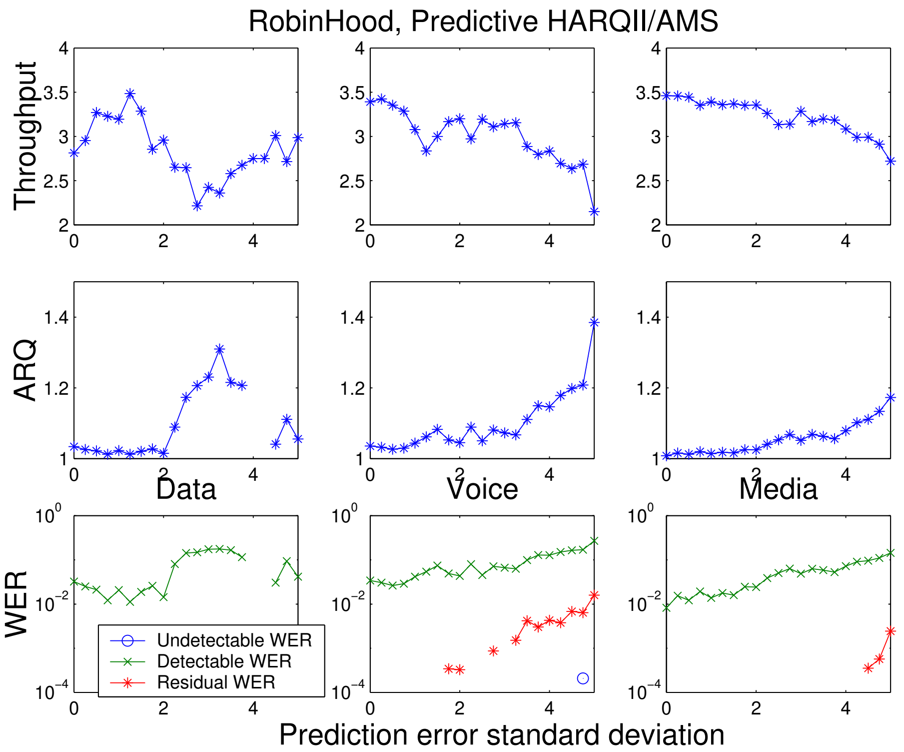

Figure 7.9: Throughput (top), ARQ effort (middle), and word error rates (bottom) when transmitting at a rate decided from prediction of channel quality, and having a Hybrid type-II ARQ improving the performance of the link layer. The prediction error standard deviation is allowed to vary along the x-axis of the plots. The ARQ effort is defined as the number of transmissions done by the link layer for the same piece of source data. The detectable word error rate can be compensated for by means of re-transmissions, which is reflected in the increased ARQ effort for higher prediction error standard deviation.

Chapter 8

Conclusions and Future Work

The thesis has treated wireless and mobile connectivity to the Internet. Problems arising in such a scenario, such as the varying quality of the radio signal, and allocation of the shared resources to different packet flows, have been discussed. A suggested solution, that applies to both problems, has been outlined and evaluated through simulations.

The suggested solution is based on channel quality predictions, that are used in combination with service requirements from different packet flows, to schedule transmissions in a spectrally efficient, and user satisfying way.

Preliminary studies of different algorithms to provide such schedules, have been made. It was found that the Robin Hood algorithm (see Section 6.5.5 on page 50) gives a very computationally attractive algorithm for time-slot scheduling. It was also found that the CSD (Controlled Steepest Descent) algorithm approximately minimizes the unfairness criteria (Equation 6.5). The CSD algorithm is, however, slightly too computationally demanding to implement in its current version.

Other network related topics have also been studied in the literature, and a summary of ongoing projects and efforts solving the wireless and mobility issues, are presented in Chapters 3 through 5.

8.1 Conclusions

A resource scheduler for the link layer transmissions is desired for spectral efficiency reasons. It was concluded in Section 7.3.2 from Figure 7.9, that a predictor with an error standard deviation below $2dB$ can achieve a residual link layer error rate, resulting in an acceptable segment error rate at the

TCP layer [37]. However, when only adaptive modulation is used together with the predictive scheduler, as in Figure 7.8, a too high segment error rate is obtained, even for link layer word error rates below 10^{-2}, that for the DATA traffic class occurs at a prediction error variance of about $2dB$. It is apparent that the introduction of Hybrid type-II ARQ significantly improves the link layer performance.

For a $1dB$ prediction error variance, with current realizations of the channel predictor, a prediction range of $0,4\lambda$ (half a wavelength) can be obtained, which at a speed of $36km/h$ and a radio frequency of $1880MHz$ corresponds to $6ms$ (or $3ms$ at $70km/h$) [56].

This motivates the introduction of an ARQ mechanism in the studied system, especially a Hybrid type-II ARQ mechanism, that handles the cases when the predictor fails to deliver accurate predictions.

In the cases that UDP is used for error-insensitive applications, the result might be satisfying in spite of a missing ARQ mechanism at the link layer, when a time-slot scheduler that takes the predicted channel quality into account, is used.

The most important questions to address is whether it is worthwile to use scheduling or not, or, even more important, to what extent we gain from scheduling. The question does not have a simple answer, since the performances can be evaluated at different layers. On the link layer we gain some insight from the investigations performed in [23]. From [23] we conclude that predictive HARQ-II/AMS outperforms the Blind HARQ-II/AMS. However, in these simulations, the Blind HARQ-II/AMS was not perfectly tuned to the channel statistics. This problem requires further investigation.

For providing different service levels, it is desirable to use some packet classification method, along with a spectrally efficient resource scheduler.

8.2 Future Work

Several topics for further research have been identified during the completion of this thesis.

8.2.1 Transport Layer Implications

What are the (exact) transport layer consequences of (not) using ARQ in the link layer?

8.2.2 Higher Dimension in Scheduler

The fading patterns show high correlation when combining both time and frequency, as can be seen from Figures 8.1 through 8.3, see [15], where darker color corresponds to higher power.

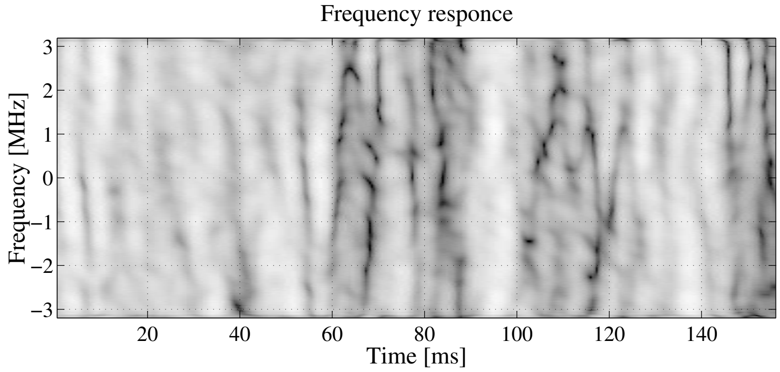

Figure 8.1: From [15]: Received power as a function of time and frequency. This channel is mainly dominated by one tap, since all frequencies fade simultaneously.

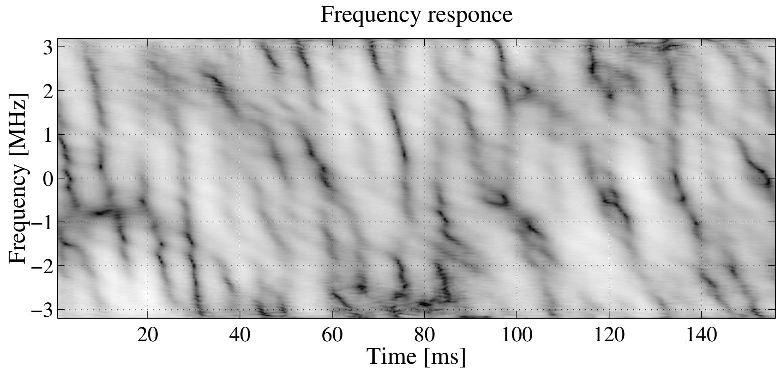

Figure 8.2: From [15]: Received power as a function of time and frequency. A clear pattern showing a chirp-behavior of the fading can be seen.

This, of course, suggests that we should utilize this second dimension too, for the scheduling approach, and not only allocate time-slots to different users. We could design an OFDM system in which we also can distribute sub-carriers over short periods of time, when the sub-carriers offer favorable conditions for high rate data transmission.

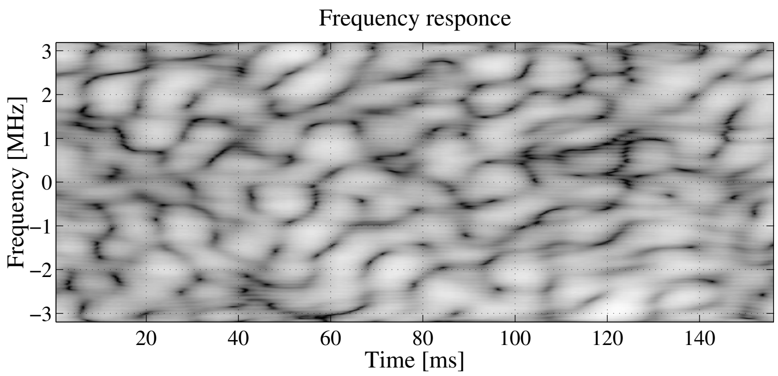

Figure 8.3: From [15]: Received power as a function of time and frequency. The coherence bandwidth is very small, since different frequencies fade independently, but if both time and frequency is considered, dependencies can be seen.

Another dimension that should be exploited, is the spatial dimension, in the sense that at one location, a mobile may have connectivity to more than one base station, thus having many independent links to choose from. This could also be exploited using prediction of channel quality and fast scheduling over many radio links. The difficulty arising here, is that many radio access points need to be involved in a more centralized resource allocation, implying an additional delay for control signaling.

8.2.3 Analytical Solution

The schedule optimization problem outlined in Appendix A.1 may have a more tractable formulation, that would allow us to find a closed form solution to the schedule optimization. Deeper analysis of the nature of the optimization problem will be the main focus of future research, in order to simplify the solution.

8.2.4 Exploit Predictor Performance Trade-Off

Since the channel predictions get worse the further into the future we look, there would be a point in trying to make the scheduling as late as possible, with as fresh predictions as possible. The scheduling could therefore be conducted in two rounds with two different prediction horizons:

1. A longer prediction horizon with less accuracy could be used to feed a long-term, first-round scheduler, that roughly assigns all time-slots

to the different users. The resulting schedule would be almost correct and optimal.

2. The second predictor works on a shorter range, with the long range predictions values as initial values, but generates more accurate predictions. The deviation from the first predictor is hopefully not too big, so then the schedule can be optimized with the more accurate predictions, making only smaller adjustments to the near-optimal original schedule.

Appendix A

Some Calculations

A.1 Lagrange Formulation of the Scheduling Problem

We here outline a re-formulation of the scheduling problem based on the criterion (6.5). Realizing that the binary constraint on the allocation matrix can be introduced using Lagrangian multipliers [40], the following set of equations would require each column in the allocation matrix to be binary and only have one “1”, the remaining elements being “0”:

$$\sum_{u=1}^{U} x_{us}^{d} = 1, \qquad s = 1, ...S, \quad d = 1...U \tag{A.1}$$

This results in S sets of U equations, that must be simultaneously satisfied within each set, to represent the binary constraint for each time-slot. The value of the exponent d reflects the number of users involved in the scheduling. For each additional user, the binary constraint must be fulfilled in a space with one additional dimension, so we need an additional linearly independent equation to reflect this increase in dimension.

This may look like a very complicated way of introducing the binary constraints, and not at all obvious at a first glance. The idea is though quite simple, which is illustrated next. In Figure A.1 this set of equations is represented geometrically for the case of one time-slot and two users. We know, in this case, that the only allowed solutions are either $(1, 0)$ or $(0, 1)$, which is where the line ($d = 1$) and the circle ($d = 2$), in Figure A.1, coincide. In the case of three users, we would instead have a three-dimensional space, with a plane for $d = 1$, a unit sphere for $d = 2$, and an infinite "wobbly" surface for $d = 3$. The system of constraint equations (A.1), would then look like

$$\left\{\begin{array}{lcl} x_{11} + x_{21} + x_{31} & = & 1 \\ x_{11}^2 + x_{21}^2 + x_{31}^2 & = & 1 \\ x_{11}^3 + x_{21}^3 + x_{31}^3 & = & 1 \end{array}\right. \tag{A.2}$$

and the solutions to (A.1) for $U = 3$ and $S = 1$, would only intersect in three points, namely $(1, 0, 0)$, $(0, 1, 0)$, and $(0, 0, 1)$. In the case of more than one time-slot, the constraints are the same, since the binary constraint in one time-slot does not affect the others.

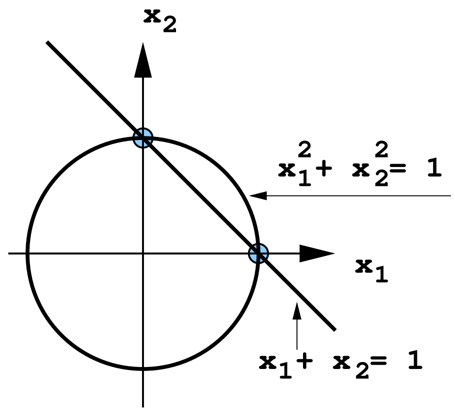

Figure A.1: Geometrical illustration of the binary constraints in two dimensions, that is, when only two users participate in the scheduling. The straight line corresponds to $d = 1$, in (A.1), and the circle to $d = 2$.

Through use of Lagrangian multipliers, the new constrained objective function becomes

$$\begin{aligned} H(x, \lambda) &= \sum_{u=1}^{U} \left(\sum_{s=1}^{S} c_{us} x_{us} - r_u \right)^2 \\ &+ \sum_{s=1}^{S} \sum_{d=1}^{U} \lambda_{ds} \left(\sum_{u=1}^{U} x_{us}^d - 1 \right) \end{aligned} \tag{A.3}$$

The problem then boils down, through calculus of variation [9], to solving a system of $2US$ non-linear (Uth order) equations, U being the number of active users, and S the number of time-slots in the scheduling window.

The $2US$ equations with the same number of unknowns (US optimization variables, x_{us}, and US Lagrange multipliers, λ_{ds}), can be reduced to US non-linear equations in the Lagrange multiplier variables. The equations have a structure that might be exploitable, namely, all the variables in an equation are of the same degree (d). The solution could be found numerically using e.g. the Newton-Raphson algorithm [50], but a more clever algorithm is sought, since the problem has an attractive structure that could be exploited. The problem might even perhaps be formulated so that a closed form solution can be found.

The final solution to the problem with this formulation is left for further research, except for the Controlled Steepest Descent (CSD) algorithm, described in Section 6.5.4, that can be viewed as an approximate numerical search solution.

A.2 Computational Complexity of the Controlled Steepest Descent Algorithm

Is the CSD algorithm too complex for a practical implementation? An approximate calculation of the required computing power is carried out in this section. In the simulated examples in Section 7.1, nine users were scheduled onto 48 time-slots. In the most naïve implementation, $(U-1) \times S \times S = 8 \times 48 \times 48 = 18432$ scalar multiplications (one evaluation of (6.5) requires $S = 48$ multiplications) are performed for each step. The algorithm takes on average 20 steps to converge (see Figure 7.4, bottom), but to be on the safe side, let's assume that 40 steps are needed. This means that $40 \times 18432 = 737280$ multiplications have to be carried out between two scheduling intervals. (For comparison, an exhaustive search would require $U^S = 9^48 \approx 6.4 \times 10^{45}$ evaluations of (6.5).) A scheduling interval is approximately $5ms$, so the computer that carries out the scheduling needs to make the calculation in approximately $2ms$ (some time is needed for other processing), resulting in a required performance of $737280/0.002 \approx 369 \cdot 10^6$ multiplications per second (369MFlops), which is easily carried out by an ordinary, fairly pipelined, 400 MHz CPU.

Reductions in complexity can be carried out, so that only 4 multiplications (by just calculating the change in (6.5), and keeping each time-slot's contribution in a separate variable) are performed at each candidate step in the CSD algorithm (resulting in $8 \times 48 \times 4 = 1536$ multiplications per step). For this example, it ends up in a reduction of 90% of the required computing power, or, alternatively ten times as many simultaneous schedules can be calculated in the same time.

A.3 Finding the Modulation Decision Thresholds

A tight upper bound on the symbol error probability due to Gaussian noise for M-QAM modulation is given by [51]:

$$P_M \leq 1 - \left[1 - 2Q\left(\sqrt{\frac{3E_{av}}{(M-1)N_0}}\right)\right]^2. \tag{A.4}$$

Here the average symbol energy E_{av}, the noise power, N_0, and the modulation format, M, are assumed to be known. The Gaussian cumulative distribution function, $Q(x)$, can be calculated according to:

$$Q(x) = \frac{1 - \mathrm{erf}(\frac{x}{\sqrt{2}})}{2}, \tag{A.5}$$

where $\mathrm{erf}(x)$ is the error function:

$$\mathrm{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt. \tag{A.6}$$

By solving (A.4) for $\frac{E_{av}}{N_0}$ and using (A.5) and (A.6), we obtain the SNIR required for a certain symbol error probability, P_M, and a given M:

$$\frac{E_{av}}{N_0} \geq \frac{2(M-1)}{3}\left[\mathrm{erf}^{-1}\left(\sqrt{1-P_M}\right)\right]^2. \tag{A.7}$$

A similar approach can be conducted for M-PSK modulations, but the symbol error probabilities are then given by

$$P_2 = Q\left(\sqrt{\frac{2E_b}{N_0}}\right) \tag{A.8}$$

for $M = 2$ (BPSK), and

$$P_4 = 2Q\left(\sqrt{\frac{2E_b}{N_0}}\right)\left[1 - \frac{1}{2}Q\left(\sqrt{\frac{2E_b}{N_0}}\right)\right] \tag{A.9}$$

for $M = 4$ (QPSK), instead.

Appendix **B**

The OSI Reference Model

Systems for communications over open networks, such as the Internet, need well-defined protocols for the communicating nodes to be able to interoperate. The protocol is a language and a set of rules to enable this understanding between nodes. In a network such as the Internet, that grew up from interconnecting a number of large wide area networks (WANs), and connecting them to smaller metropolitan and local area networks (MANs and LANs), it is not only the end nodes that might span a large spectrum of complexity and level of abstraction. Also the sub-networks that are the building blocks on the way, may have different topologies and internal protocols, since they appeared independently and were interconnected afterwards. For this reason, there is a large heterogeneity among different vendors', and net-owners' ways of implementing the communication between nodes in their networks. In general, this is not a problem thanks to the layered structure of the OSI Reference Model, see Figure B.1.

Layers in the OSI Model represent different levels of abstraction in the communication system. The top layer might be the most familiar one to a user. This is called the Application Layer, and it is here we can find such protocols as the hyper-text transfer protocol (HTTP), and network file system (NFS). They represent the highest level of abstraction, where a user can talk about transferring files, and downloading images. At the other end of the OSI Model we have the Physical Layer. This is where the data is translated to electrical or optical signals that are applied to the transmitting media, which could be a copper wire, an optical fiber, or an electromagnetically radiating antenna (radio). In this layer, the level of abstraction is dealing with processing of the transmitted and received physical signals, and control of access to the physical media.

Only from these two levels of abstraction it is easily seen that one single protocol would make it extremely difficult to communicate between different machines. Why should I, as a sender of a message, need to take into account what kind of networks my message has to pass through, on the way to the desired receiver? I only want to know that I'm sending an email to my friend (and maybe not even that, since my friend might prefer to have it translated to a voice mail). For this reason, there are seven layers in the OSI Reference Model: The Application, Presentation, Session, Transport, Network, Link, and Physical layers. Each of the layers communicate via a protocol with a corresponding layer in another host (the destination host or an intermediate node).

Between the different layers there are well-defined interfaces. The lower layers supply services to the higher layers through these interfaces.

To describe what the different layers do, we can follow the process of sending an email:

Application The email program reads the input along with the email address of the recipient. It checks for the existence of the recipient, and if everything is OK, then the email is transferred.

> **Request:** Translate the contents of the email to the appropriate encoding, and make sure the recipient receives it.

Presentation At this layer, conversion is done between different encodings of characters and data types from the host's representation, to the network's representation and back.

> **Request:** Transfer this data to the email program on this address.

Session The session layer initiates a session between the hosts, by first finding the destination host in the domain name server (DNS) system, and then ensures that the different processes on the different hosts connect and transfer the data.

> **Request:** Create a reliable connection to the email program on this host (IP-number).

Transport Creates a reliable connection to the recipient host, and balances the load on the network by controlling the amount of data transmitted. The data is bundled in segments, so that no segments are too big for the underlying network.

Request: Take this set of data and send it to this (IP) address.

Network Packs the data in an IP-packet and sends it off to the right branch in the network based on a routing table, usually the default gateway out of the local network.

Request: Send this packet to the router in that direction.

Link Fragments or concatenates IP-packets to link frames, so that they fit into the link layer transmission format. Also handles error protection over the link.

Request: Take this framed bit-stream and put it on the wire.

Physical Takes the link layer frames and translates them into a symbol-stream that in turn is mapped onto electrical or optical signals on the physical channel.

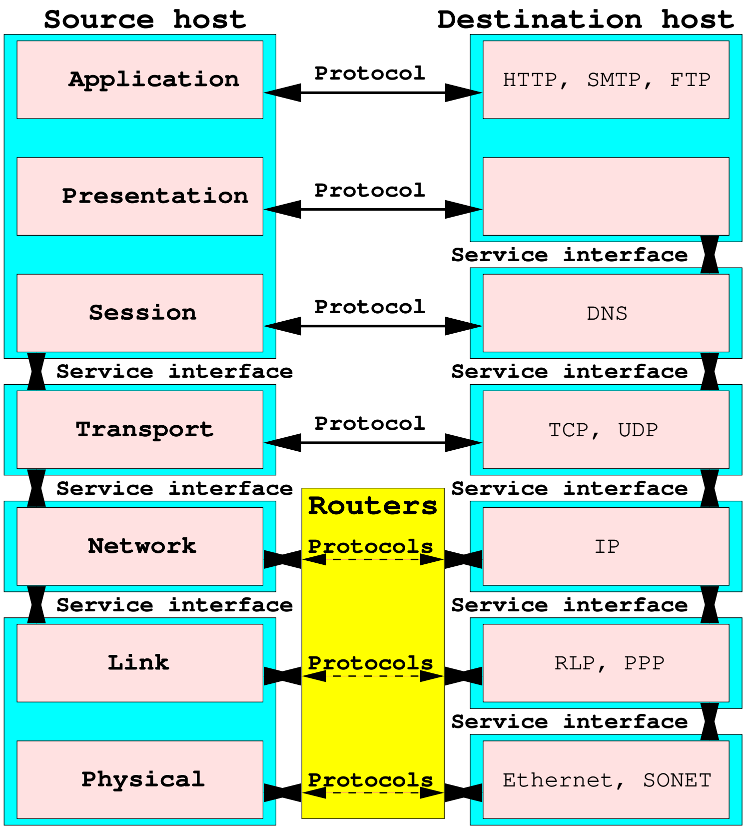

Figure B.1: The OSI reference model proclaims a layered structure for packet based communications over heterogeneous networks, allowing for different manufacturers to use their own solutions in their implementations. Not all layers need to be present, since several can be included into one single level of abstraction. To the left the names of the layers are presented, and to the right, some examples of protocols that operate in each layer are displayed.

Appendix C

Words and Acronyms

C.1 Words

access network The last part of the network, where the accessing hosts reside. To be separated from the transportation, or backbone, network.

fading Temporary loss of signal power due to radio interference or shadow.

header The part of a data packet that contains control and security information. To be separated from the content or payload.

host A machine on a network, capable of running user application programs. A mobile phone is a host and a home PC also is a host. A router is not a host.

interface The border between two layers in the protocol stack. The lower layer offers services to the higher layer, which in turn provides control information and content (payload).

jitter The deviation in the transmission delay for a packet stream. Defined as two times the number of micro-seconds that a packet can be early or late, compared to the average delay.

modulation The process of translating digital information into physically measurable and transportable signals.

payload The content of a data packet that the higher layer in the protocol stack submits for delivery. To be separated from the header.

protocol A language and a set of rules for how two parties should communicate.

proxy A host intended to make networking more efficient by replicating, or representing, a different host and its services. A web proxy can for example store popular remote web pages in a local cache memory, and access the cache instead of passing the request to the Internet. The meaning of the word "proxy" is *authority to act for another.*

router A machine that works as a network node at the network layer in a packet-switched communication system. It reads the destination IP (and source IP) from the incoming packets and delivers them on the right output port, according to a router table, so that it reaches the right host. Routers send messages between themselves about changes in the network topology.

scheduling Rules for how different queues in a queuing system should be served. A common scheduling algorithm is Round-Robin, that simply allocates resources for a pre-defined amount of time to clients in a cyclical order.

switch The circuit-switched counterpart to a router. Sets up a logical or physical circuit connection between two lines on a network.

transport network The backbone part of the network, only containing routers and high speed links. To be separated from the access network.

C.2 Abbreviations and Acronyms

3GPP Third Generation Partnership Project. A collaborative organization consisting of international standard bodies within telecommunications and computer communications.

ACK Acknowledgment. Sent by a receiver of a message if the message seems to be correct. In TCP, duplicate ACKs can be sent when messages arrive out of order, announcing a missing expected segment, which can be interpreted as a NAK.

ARQ Automatic Repeat reQuest. A mechanism that handles repetition of erroneously received data. Comes in many flavors with varying features and complexity.

ATM Asynchronous Transfer Mode. Link layer protocol where data is sent in small cells (packets).

BER Bit Error Rate. Expressed as the ratio between erroneous bits and total number of bits.

BPSK Binary Phase Shift Keying. PSK with two possible symbols, resulting in a rate of one bit per symbol ($M = 2, R = 1$).

CDMA Code Division Multiple Access. Method for multiplexing many communication channels onto one common physical channel. The channels are divided by individual codes and resolved by, for example, matched filtering.

CRC Cyclic Redundancy Check. A special cyclic block code for fast error detection.

DTM Dynamic synchronous Transfer Mode. Link layer protocol where data is sent in dynamically allocated time-slots over high speed optical links.

FDD Frequency Division Duplex. Method for multiplexing transmissions in two directions over the same physical media. In FDD, the two communicating nodes transmit at two different carrier frequencies, avoiding collisions.

FEC Forward Error Correction. Link layer functionality for protection against erroneous reception of transmitted data. Has ability to discover and correct errors in the received data.

GSM Global System for Mobile communications. Standard for second generation mobile communications.

IETF Internet Engineering Task Force. A community of experts concerned with the evolution of the Internet.

IP Internet Protocol. Network layer protocol for communication on the Internet. The current version of IP is called IPv4, and work is on-going on the development of version 6, IPv6, with a richer functionality and a larger address space.

MPEG Moving Pictures Expert Group. Standardization group for transmission and storage of digital images and sound.

MSS Maximum Segment Size. The maximum number of bytes that TCP will put in a TCP segment. This number is negotiated between the sender and the receiver (the minimum of the two options).

NAK Negative Acknowledgment. Sent by a receiver of a message if the message seems to be in error or missing.

OFDM Orthogonal Frequency Division Multiplexing. Method for transmitting over densely separated frequency channels in parallel.

PSK Phase Shift Keying. Modulation method where information is stored in the phase of the transmitted signal.

QAM Quadrature Amplitude Modulation. Amplitude modulation method where amplitudes in two orthogonal phase angles are used.

QoS Quality of Service. Practical buzz-word referring to the problem of attaining certain service levels in best-effort packet networks, by adding rules to distinguish different packets, and rules for how these different packets should be served.

QPSK Quaternary Phase Shift Keying. PSK with four possible symbols, resulting in rate of two bits per symbol ($M = 4, R = 2$).

RTCP RTP Control Protocol.

RTP Real-Time Protocol. Preferred application protocol for real-time (interactive) communications.

RTT Round-Trip Time. A state variable in TCP, representing the time it takes for a packet to reach the destination and the ACK to return to the sender.

SONET Synchronous Optical NETwork. Optical backbone physical layer protocol, originally used for telecommunications.

TCP Transmission Control Protocol. Transport layer protocol for reliable delivery of data.

TDD Time Division Duplex. Method for multiplexing transmissions in two directions over the same physical media. In TDD, the two communicating nodes take turns in transmitting, avoiding collisions.

UDP User Datagram Protocol. Transport layer protocol for fast, unreliable delivery of data.

UMTS Universal Mobile Telecommunications System. Third generation mobile communication standard.

UTRAN UMTS Terrestrial Radio Access Network.

WCDMA Wide-band CDMA. A standardized radio interface for UMTS.

Bibliography

[1] L. Ahlin and J. Zander. *Principles of Wireless Communications.* Studentlitteratur, Lund, Sweden, 2 edition, 1998.

[2] M. Allman, V. Paxson, and W. Stevens. TCP Congestion Control. RFC2581, Apr. 1999.

[3] H. Balakrishnan, S. Seshan, E. Amir, and R. H. Katz. Improving TCP/IP Performance over Wireless Networks. In *Proc. MOBICOM*, pages 2–11, Berkeley, CA, 1995.

[4] C. F. G. Bispo, J. ao J S Sentieiro, and R. D. Hibberd. Adaptive Scheduling for High-Volume Shops. *IEEE Trans. on Robotics and Automation*, 8(6):696–706, Dec. 1992.

[5] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. An Architecture for Differentiated Services. RFC2475, Dec. 1998.

[6] J. Border, M. Kojo, J. Griner, G. Montenegro, and Z. Shelby. Performance Enhancing Proxies. Internet Draft, Nov. 2000. http://www.ietf.org/internet-drafts/draft-ietf-pilc-pep-05.txt.

[7] R. Braden, D. Clark, and S. Shenker. Integraed Services in the Internet Architecture: an Overview. RFC1633, June 1994.

[8] A. Brunström and T. Ottosson. The Introduction of Soft Information in IP-based Wireless Networks. In *Nordic Radio Symposium*, Nynäshamn, Sweden, Apr. 2001.

[9] A. E. Bryson and Y.-C. Ho. *Applied Optimal Control.* Hemisphere Publishing Company, 1975.

[10] J. Carlström. *Reinforcement Learning for Admission Control and Routing.* PhD thesis, Uppsala University, May 2000.

[11] S.-G. Chua and A. Goldsmith. Variable-Rate variable-power MQAM for Fading Channels. In *VTC*, pages 815–819, Atlanta, Georgia, May 1996.

[12] M. E. Crovella and A. Bestavros. Explaining World Wide Web Traffic Self-Similarity. Technical report, Computer Science Department, Boston University, 1995. Revised.

[13] S. Deng, A. R. Bugos, and P. M.Hill. Design and evaluation of an ethernet-based residential network. *JSAC*, 14(6):1138–1150, August 1996.

[14] A. Drukarev and D. J. C. Jr. Hybrid ARQ error control using sequential decoding. *IEEE Transactions on Information Theory*, IT-29:521–535, July 1983.

[15] T. Ekman. *Prediction of Mobile Radio Channels*. Licentiate Thesis, Uppsala University, Dec. 2000.

[16] T. Ekman and G. Kubin. Nonlinear prediction of mobile radio channels: Measurements and MARS model designs. In *ICASSP*, Phoenix, Arizona, March 1999.

[17] N. C. Ericsson. Adaptive modulation and scheduling for fading channels. In *GLOBECOM*, Rio de Janeiro, Brazil, December 1999.

[18] N. C. Ericsson. Adaptive Modulation and Scheduling of IP Traffic over Fading Channels. In *Proc. IEEE Vehicular Technology Conference*, pages 849–853, Amsterdam, the Netherlands, Sept. 1999.

[19] N. C. Ericsson, A. Ahlén, S. Falahati, and A. Svensson. Hybrid type-II ARQ/AMS supported by Channel Predictive Scheduling in a Multi-User Scenario. In *Proc. IEEE Vehicular Technology Conference*, Boston, Massachusetts, Sept. 2000.

[20] M. Eriksson, A. Furuskär, M. Johansson, S. Mazur, J. Molnö, C. Tidestav, A. Vedrine, and K. Balachandran. The GSM/EDGE Radio Access Network - GERAN; System Overview and Performance Evaluation. In *Proc. IEEE Vehicular Technology Conference*, pages 2305–2309, Tokyo, Japan, May 2000.

[21] A. Ewerlid. Reliable Communication over Wireless Links. In *Proc. PCC Workshop*, Nynäshamn, Sweden, Apr. 2001.

[22] S. Falahati. Convolutional Codes for Wireless Packet Data Systems. Licentiate Thesis, Chalmers University of Technology, Feb. 2000.

[23] S. Falahati and N. C. Ericsson. Hybrid type-II ARQ/AMS and Scheduling using Channel Prediction for Downlink Packet Transmission on Fading Channels. In *Proc. PCC Workshop*, Nynäshamn, Sweden, Apr. 2001.

[24] S. Falahati, T. Ottosson, A. Svensson, and L. Zihuai. Convolutional Coding and Decoding in Hybrid type-II ARQ Schemes on Wireless Channels. In *Proc. IEEE Vehicular Technology Conference*, pages 2219–2224, Houston, Texas, May 1999.

[25] S. Falahati, T. Ottosson, A. Svensson, and L. Zihuai. Hybrid type-II ARQ Schemes based on Convolutional Codes in Wireless Channels. In *Proc. FRAMES Workshop*, pages 225–233, Delft, the Netherlands, Jan. 1999.

[26] S. Falahati and A. Svensson. Hybrid type II ARQ schemes for Rayleigh fading channels. In *Proc. International Conference on Telecommunications*, volume 1, pages 39–44, Porto Carras, Greece, June 1998.

[27] K. Fall and S. Floyd. Simulation-based Comparisons of Tahoe, Reno, and SACK TCP. Technical report, Lawrence Berkeley National Laboratory, Dec. 1995. ftp://ftp.ee.lbl.gov/papers/sacks.ps.Z.

[28] F. Freiha, K. Chandra, V. Mehta, and C. Thompson. Performance of VBR Video With Equalization on Wireless Fading Channels. In *Proc. IEEE Global Telecommunications Conference*, pages 2642–2647, Rio de Janeiro, Brazil, Dec. 1999.

[29] A. Furuskär, S. Mazur, F. Müller, and H. Olofsson. EDGE: enhanced data rates for GSM and TDMA/136 evolution. *IEEE Personal Communications*, 6(3):56–66, June 1999.

[30] A. Furuskär, J. Näslund, and H. Olofsson. EDGE: Enhanced data rates for GSM and TDMA/IS-136 evolution. Ericsson Review, Jan 1999.

[31] D. G-Kurup. Radio Frontend Integration Techniques: CAD Models and Applications. Licentiate Thesis, Uppsala University, Dec. 2000.

[32] F. S. Hillier and G. J. Lieberman. *Introduction to Operations Research*. McGraw-Hill Book Co., Singapore, 1990.

[33] D. Huang and J. J. Shi. Performance of TCP Over Radio Link With Adaptive Channel Coding and ARQ. In *Proc. IEEE Vehicular Technology Conference*, volume 3, pages 2084–2088, Houston, Texas, May 1999.

[34] G. Huston. Next Steps for the IP QoS Architecture. RFC2990, Nov. 2000.

[35] L.-E. Jonsson, M. Degermark, H. Hannu, and K. Svanbro. RObust Checksum-based header COmpression (ROCCO). Internet Draft, June 2000.

[36] M. Kawagishi, S. Sampei, and N. Morinaga. A Novel reservation TDMA Based Multiple Access Scheme using Adaptive Modulation for multimedia Wireless. In *Proc. IEEE Vehicular Technology Conference*, pages 112–116, May 1998.

[37] H. Kruse, S. Ostermann, and M. Allman. On the performance of TCP-based data transfers on a faded Ka-band satellite link. In *Ka-Band Utilization Conference*, Cleveland, Jun 2000.

[38] E. Lindskog. *Space-Time Processing and Equalization for Wireless Communications*. PhD thesis, Uppsala University, May 1998.

[39] S. Lu, V. Bharghavan, and R. Srikant. Fair Scheduling in Wireless Packet Networks. *IEEE/ACM Transactions on Networking*, 7(1):10–22, Feb. 1999.

[40] D. G. Luenberger. *Linear and Nonlinear Programming*. Addison-Wesley Publishing Company, 1984.

[41] S. Mascolo, C. Casetti, M. Gerla, S. S. Lee, and M. Sanadidi. TCP Westwood: congestion control with faster recovery. Technical Report 200017, Computer Science Department, UCLA, Los Angeles, CA, 2000.

[42] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanov. TCP Selective Acknowledgement Options. RFC2018, Oct. 1996.

[43] D. Mitzel. Overview of 2000 IAB Wireless Internetworking Workshop. RFC3002, Dec. 2000.

[44] J. R. Moorman and J. W. Lockwood. Implementation of the Multiclass Priority Fair Queueing (MPFQ) Algorithm for Extending Quality of Service in Existing Backbones to Wireless Endpoints. In *Proc. IEEE Global Telecommunications Conference*, pages 2752–2757, Rio de Janeiro, Brazil, Dec. 1999.

[45] M. Naijoh, S. Sampei, N. Morinaga, and Y. Kamio. ARQ Schemes with Adaptive Modulation/TDMA/TDD Systems for Wireless Multimedia Communication Services. In *Proc. IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, volume 2, pages 709–713, Helsinki, Finland, Sept. 1997.

[46] T. Öberg. *Modulation, detektion och kodning*. Studentlitteratur, 1998.

[47] A. Olsson, editor. *Understanding Telecommunications*, volume 1, 2. Ericsson Telecom, Telia and Studentlitteratur, 1998.

[48] C. Perkins. IP Mobility Support. RFC2002, Oct. 1996.

[49] J. Postel. Transmission Control Protocol. RFC793, 1981.

[50] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. *Numerical Recipes - The Art of Scientific Computing*. Cambridge University Press, 1989.

[51] J. G. Proakis. *Digital Communications*. McGraw-Hill, New York, 3rd edition, 1995.

[52] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RTP: A Transport Protocol for Real-Time Applictions. RFC1889, Jan. 1996.

[53] H. Schulzrinne, A. Rao, and R. Lanphier. Real Time Streaming Protocol (RTSP). RFC2326, Apr. 1998.

[54] A. Shacham, R. Monsour, R. Pereira, and M. Thomas. IP Payload Compression Protocol (IPComp). RFC2393, Dec. 1998.

[55] T. Socolofsky and C. Kale. A TCP/IP Tutorial. RFC1180, Jan. 1991.

[56] M. Sternad, T. Ekman, and A. Ahlén. Power prediction on broadband channels. In *Proc. IEEE Vehicular Technology Conference*, Rhodes, Greece, May 2001.

[57] M. Sternad, A. Svensson, A. Ahlén, and T. Ottosson. PCC Wireless IP - Optimizing Throughput and QoS over Fading Channels. In *Nordic Radio Symposium*, Nynäshamn, Sweden, Apr. 2001.

[58] A. S. Tanenbaum. *Computer Networks*. Prentice Hall International, Upper Saddle River, New Jersey, 3 edition, 1996.

[59] Technical Specification Group - Radio Access Network. UTRA High Speed Downlink Packet Access. Technical Report 3GPP TR 25.950 version 2.0.0, 3rd Generation Partnership Project (3GPP), 2001.

[60] Technical Specification Group - Services and System Aspects. QoS Concept and Architecture. Technical Report 3G TR 23.907 version 1.6.0, 3rd Generation Partnership Project (3GPP), 1999.

[61] C. Tidestav. *The Multivariable Decision Feedback Equalizer - Multiuser Detection and Interference Rejection*. PhD thesis, Uppsala University, Dec. 1999.

[62] T. Ue, S. Sampei, and N. Morinaga. Symbol Rate and Modulation Level Controlled Adaptive Modulation/TDMA/TDD for Personal Communication Systems. In *Proc. IEEE Vehicular Technology Conference*, volume 1, pages 306–310, Chicago, Illinois, July 1995.

[63] T. Ue, S. Sampei, and N. Morinaga. Adaptive Modulation Packet Radio Communication System using NP-CSMA/TDD Scheme. In *Proc. IEEE Vehicular Technology Conference*, pages 416–420, Atlanta, Georgia, May 1996.

[64] J. Walrand and P. Varaiya. *High-Performance Communications Networks*. Morgan Kauffmann Publishers, 2 edition, 2000.

[65] M. Wennström. Smart Antenna Implementation Issues for Wireless Communications. Licentiate Thesis, Uppsala University, Oct. 1999.

[66] S. B. Wicker. *Error control systems for digital communication and storage*. Prentice-Hall, Englewood Cliffs, NJ, 1995.