Nilo Casimiro Ericsson

Revenue Maximization in Resource Allocation: Applications in Wireless Communication Networks

October 2004

SIGNALS AND SYSTEMS
UPPSALA UNIVERSITY
UPPSALA, SWEDEN

Submitted to the Faculty of Science and Technology, Uppsala University in fulfillment of the requirements for the degree of Doctor of Philosophy.

UPPSALA
UNIVERSITET

ISBN 91-506-1773-7
Printed in Sweden by Eklundshofs Grafiska, Uppsala, 2004

Till min ängel

Abstract

Revenue maximization for network operators is considered as a criterion for resource allocation in wireless cellular networks. A business model encompassing service level agreements between network operators and service providers is presented. Admission control, through price model aware admission policing and service level control, is critical for the provisioning of useful services over a general purpose wireless network. A technical solution consisting of a fast resource scheduler taking into account service requirements and wireless channel properties, a service level controller that provides the scheduler with a reasonable load, and an admission policy to uphold the service level agreements and maximize revenue, is presented.

Two different types of service level controllers are presented and implemented. One is based on a scalar PID controller, that adjusts the admitted data rates for all active clients. The other one is obtained with linear programming methods, that optimally assign data rates to clients, given their channel qualities and price models.

Two new scheduling criteria, and algorithms based on them, are presented and evaluated in a simulated wireless environment. One is based on a quadratic criterion, and is implemented through approximative algorithms, encompassing a search based algorithm and two different linearizations of the criterion. The second one is based on statistical measures of the service rates and channel states, and is implemented as an approximation of the joint probability of achieving the delay limits while utilizing the available resources efficiently.

Two scheduling algorithms, one based on each criterion, are tested in combination with each of the service level controllers, and evaluated in terms of throughput, delay, and computational complexity, using a target test system. Results show that both schedulers can, when feasible, meet explicit throughput and delay requirements, while at the same time allowing the service level controller to maximize revenue by allocating the surplus resources to less demanding services.

Acknowledgments

First of all, my supervisors Professor Anders Ahlén and Professor Mikael Sternad, deserve my greatest gratitude for sharing their enthusiasm, knowledge, and experience, but also for pushing me forward in times of less progress. It has been fun and interesting to work with you, and I hope that we continue our work together in the same familiar spirit. Thanks also to all the people in the Signals & Systems group at Magistern, that also contribute to the stimulating atmosphere, and especially to Mathias Johansson for also proof-reading parts of this thesis.

The work is supported by the Swedish Foundation of Strategic Research, through the PCC (Personal Computing and Communications) research program. Within PCC, the Wireless IP (WIP) project develops innovative approaches to increase spectrum efficiency and throughput for data over wireless links.

Family and friends make the effort worthwile. I'm glad I have you to share the good times, and the not so good times, with.

Finally, I want to acknowledge the huge support from Jenny, my partner for life. There has to be a meaning with the things that you do every day, and to me, Jenny has brought that, and much more.

For you who read my licentiate thesis -

here's the sequel, and it's better!

Contents

Chapter 1

Introduction

We consider the problem of distributing a limited amount of shared resources among a number of clients, in a fashion that optimizes a revenue-based criterion. More specifically, we consider the problem of service *resource scheduling* to a population of clients with different and time-varying *service requirements* and also different and time-varying resource utilization per service unit. Furthermore, the clients generate different *revenue* for the owner of the server. The question we try to answer is: How do we decide who should use the shared resource when?

This problem is found in wireless mobile communications, where different *mobile hosts* are travelling at different speeds and directions, and at different distances to a radio signal transmitting *base station*. The mobile hosts therefore experience different and varying *signal qualities*, affecting the capacity[1] of the resource they utilize for transmission of information. Since different users may run different applications on the mobile hosts, they also have varying *service demands*.

Similar problems are encountered in areas where scheduling is used as a tool for maximizing some measure of efficiency, as in a common *workshop scheduling* problem (a number of production machines with limited capacity should be used for tasks on different products, maximizing profit), or in a *processor sharing* multi-user computer system (a computer processor is used for multiple jobs lined up in a queue, minimizing e.g. waiting time). There is one key difference between our current scheduling problem and problems

[1]The term *capacity* does not refer to Shannon capacity [68] in this thesis. Our meaning of capacity refers to the service level that can be achieved per unit resource, and is termed *bin capacity*, as defined later in Definition 4.1.

previously described and (sometimes) solved: The service level obtained per resource utilization (the capacity) will be different for different clients, and also varying.

The criterion to be maximized has been chosen to be the *profit* of the wireless network operator. However, since the approach in this thesis only helps to increase income by successful operation (not to control overall cost), *profit* has been replaced by *revenue* in the continuation.

To illustrate the effects of resource scheduling on the revenue, we introduce a business model where *service providers*, or *content providers*, buy the wireless access to their *end users* from the wireless *network operators*. In this business model, the service providers sign *service level agreements* with a network operator. In order to generate maximal revenue for the network operator, the resources will have to be utilized efficiently, perhaps by overbooking them, and dynamically allocating them to the clients that pay the most for them. This allocation is performed by the resource scheduler.

Besides from scheduling, *admission control* that operates in the light of the service level agreements and the corresponding *price models*, is introduced in order to assign the resources to efficiently serve the clients, but also to take action when the overbooked resources become overloaded.

1.1 Contributions and Outline

The thesis spans a wide research area. The base is built on the business assumption that services have an economic value for its clients, and that the value has to be larger than the cost, in order for the provision of services to be worthwile. The coupling of price models to the wireless resource allocation problem is the first contribution of this thesis, and is mainly presented in Chapter 2.

The second contribution is that of proposing four new on-line scheduling algorithms that are aware of both the resource availability and the service requirements. Three of the algorithms are based on a quadratic criterion, whose minimization assigns the available resources to efficiently meet the service requirements. The fourth algorithm uses a probabilistic criterion that combines the maximization of the probability of utilizing a resource while it gives good capacity, with the minimization of the probability of failing to meet a client's service requirements. The algorithms are presented in Chapter 6 and Chapter 7, with a preceding discussion of the requirements on such algorithms, in Chapter 4.

The third contribution is that of proposing methods for coupling the price models and the service level agreements to both long term, and short term, resource allocation methods. This is achieved by means of a novel way of regarding admission control as two entities, namely, admission policing, and service level control. The contribution is mainly within automatic control of assigned service levels, in order to adjust them to the available resources, in the light of the existing price models and service level agreements. This is presented in Chapter 3.

The final contribution is that of applying the proposed methods on a test system, that candidates as a future mobile wireless communication network solution. This is mainly in the form of simulations presented in Chapter 9, but also as a discussion of the requirements and the applicability of the proposed methods, in Section 4.3.

The thesis is concluded with suggestions for future work in Chapter 10.

The remainder of Chapter 1 will serve as an introduction to the ideas pursued in the thesis, but also provide some background material from the data communications area.

1.2 Mapping Application Requirements onto Service Requirements

In the previous section we argued that service requirements vary depending on the type of application that is running on the communicating hosts. The application requirements are often expressed at a high level, such as "good speech perception", "low dialogue delay", and "simple email and web browsing". These requirements need to be mapped into low-level service parameters that can be quantified.

At the low level there are mainly four characteristics that can be controlled by means of admission control and scheduling. These are:

Throughput, described by e.g. a Token Bucket, see Definition 1.1.

Delay, which may be approximately translated into a corresponding queue size, given the throughput above, by means of Little's formula [44].

Data Loss statistics. Different applications are differently sensitive to loss of data. Therefore different services could accept different data loss rates. For example real-time multimedia conversations accept more data loss than file transfers, and may therefore accept a higher target error rate.

Admission. A connection can be established or released for different reasons. One connection can be replaced by another if circumstances allow or require it.

In the 3GPP specifications for 3G wireless services [81, 80], much effort has been spent on defining parameters with different target values for different service classes. This mapping is not a trivial one, and the way it is performed will distinguish different service providers from one another.

We will in this thesis focus on the admission, the throughput, and the delay characteristics of a transmission service. The data loss statistics are only introduced indirectly, through the achievable transmission rate, given a certain target error rate and the wireless channel quality.

A central concept in our presentation is the *token bucket*, defined below. However, we will use it in a slightly modified version, as described in Remark 1.1.

Definition 1.1: *Token Bucket [22]*

A *token bucket* is a dynamic method for shaping data traffic in terms of a *persistent transmission rate*, and a *maximum burst size*. A *token* represents a right to transmit a certain amount of data, and the token is consumed when that amount of data has been transmitted. The token bucket can store such tokens for later use, and its dynamics is governed by the following two parameters:

- The *token rate* defines the rate at which the token bucket is filled with new tokens, and it represents the persistent transmission rate that a data flow can maximally maintain.

- The *bucket size* defines the number of tokens that a bucket may contain, and it represents the maximum amount of data that a data flow may momentarily transmit (the maximum burst size).

Data that has been transmitted, having the corresponding tokens in the bucket, is said to be *conformant*, whereas data transmitted *without* having the corresponding tokens in the bucket is said to be *non-conformant*.

■

A bucket starts up filled (up to the bucket size) with tokens. For each data packet that passes the bucket, a number of tokens corresponding to the packet size are removed from the bucket. The bucket is then re-filled with new tokens according to its token rate, until it reaches its bucket size.

Remark 1.1: *Tokens represent service units*

We will utilize the *token* abstraction mainly for the purpose of controlling the service levels of different clients. We have therefore chosen to express service units in terms of tokens. Tokens will be granted to clients in a similar fashion as that of a token bucket (see Definition 1.1 above), but we will introduce the possibility to adjust the token rates, and also to control how and when the tokens are spent. A token is regarded as a *granted right to receive a certain amount of a service.*

■

1.2.1 Internet Protocol

It is envisioned that all communication will sooner or later be carried over the Internet Protocol (IP). IP is the protocol providing functionality for data packets finding their way, through the network, hop by hop, to the destination. It has the potential advantage of being a distributed packet-based protocol for flexible and robust forwarding of data, using relatively cheap equipment. There are still some weaknesses that will need to be removed in order to enable the conversational services that wireless cellular telephony systems offer. One weakness is the support for mobility and hand-over functionality, that still is too slow to handle conversational service quality. Another weakness is the large overhead associated with IP traffic: Since IP is packet switched (no established end-to-end connection), each packet must carry header information about its source, destination, service parameters, ordering number, etc. The overhead becomes a problem at the wireless link, where transmission resources need to be efficiently utilized.

Both the mentioned problems are currently being addressed by the Internet Engineering Task Force (IETF), see e.g. [20] for header compression, and [52] for Mobile IP handover optimization.

Transmission Control Protocol

Transmission Control Protocol (TCP) performance is very sensitive to variations in the underlying link layer. Throughput and delay are tightly coupled when using this transport protocol due to the transmission and congestion control mechanisms built into TCP [2, 64, 70, 77]. Variations in throughput lead to perceived variations in delay, and variations in delay lead to perceived loss of data, that in turn generates excess load by retransmitting assumedly lost data. Many suggestions on how to improve TCP to cope with the variations have been presented over the years [9, 19, 39, 53, 77].

Some have been more successful than others and an observation is that the fact that TCP protocols reside in the end hosts, and that they need to follow a common protocol, makes it difficult to agree upon a common standard for TCP improvement over wireless links. Thus, to avoid the problems related to triggering of retransmissions, it is important that hosts running TCP communications over a wireless channel perceive a stable service in terms of throughput and delay.

User Datagram Protocol

It is more difficult to draw any general conclusions on how UDP based communications react to wireless transmissions of varying quality. UDP does not follow any common rules of behaviour to different traffic situations since it is up to the application programmer to handle reliability issues by implementing protection against jitter, packet loss, and reordering [63, 66, 43].

1.2.2 Service Level Control in IP: IntServ and DiffServ

There are two directions for service level control within the Internet community: Differentiated services (DiffServ) [17], and integrated services (IntServ) [21].

The DiffServ standard defines service *classes* and their corresponding desired service parameters, leaving the resource allocation to be controlled by each transmission node on the way, based on the corresponding service class. IntServ, on the other hand, explicitly reserves all the required resources when setting up the connection between the communicating hosts.

For our proposed methods, the DiffServ approach is preferred. Our service reuirements are defined per class, but provided per flow by means of channel quality dependent, and revenue dependent, traffic shaping. This is possible since we are handling an *access router*[2], the base station, with explicit knowledge of the end host and its SLA-memberships.

1.3 Resources and Capacity

The available physical resources to provide the services outlined above are under our assumptions limited to radio spectrum. Other resources, such

[2]Compare with [85], dealing with edge and core routers, that do not have information about the end hosts, and therefore require additional communication overhead in order to provide this information.

as electrical power for transmission, reception, and processing, are omitted from further discussion in this thesis. Furthermore,

- we regard the resources as fixed portions of a fixed global resource pool,
- the resources are *shared* between a number of resource consumers, or clients, so that a specific piece of the resource that is available for many consumers, can only be consumed by one of them. Thus, the clients are mutually exclusive consumers of a piece of the resource.
- The resources cannot be stored for later consumption. They have to be consumed as soon as they arise, since they will otherwise be forfeited.
- Different resource portions will offer different and varying capacities.
- Furthermore, different consumers will be able to utilize the resources differently, achieving different capacities.

These variations in capacity are due to the varying channel qualities perceived by different users. They originate mainly from radio physical phenomena known as *path loss*, *shadow fading*, and *small-scale fading*, in decreasing order of time scale, and length scale. The scheduling algorithms that we will present later in the thesis are designed to exploit these variations.

1.3.1 Partition of the Resources

In order to individually allocate the available radio resources to different resource consumers, we wish to partition them into small portions, such that they can be utilized when and where they offer the best capacity. In order to achieve this, the size of the resource portions should be chosen such that we can exploit also the fast fading of the received power for mobile users, due to the small-scale fading of the channels.

A natural subdivision of the resources is achieved by partitioning them in time, in frequency, and in space.

Resource Partitioning in Time

Channel properties will change over time. The spectrum may be divided into time slots so that we can exploit the time variations. The appropriate duration of a time slot may be calculated from the channel's *coherence time*, that depends on the channel impulse response, that in turn depends on the speed at which a receiver or transmitter moves.

In this work, we assume that the channel parameters in consecutive time slots may be correlated, but that there is no inter-slot interference. Thus, the transmission in earlier slots does not interfere with the transmission in later slots.

Resource Partitioning in Frequency

Similarly to the time scale, there is a variation of channel properties over frequency, given by the *coherence bandwidth*, whithin which a channel is correlated to a certain level. By dividing a frequency band into sets of subcarriers, sized narrower than the coherence bandwidth, we obtain a subdivision into resource pieces that can be allocated to exploit the variations.

Spatial Resource Partitioning

In the spatial domain, transmission over neighboring[3] resources may cause more or less interference on each other, depending on the (difference in) distance between them. They may at the same time be independent in the sense that their channel properties are different. A mobile at one location may communicate with one or several base stations or antennae in one time-frequency bin, thus either combining the different signals, or, choosing between the independent channels in the spatial domain. We take advantage of the fact that for a given point in time and frequency, resources allocated to one connection can be re-used on a location at a certain spatial distance away, where the interference is negligible.

1.3.2 Efficient Use of the Resources

The admission control described in the following section has to consider traffic of different types, generating more or less revenue for the operator. One portion of the resources should be allocated to a service that can guarantee a certain delay at a limited throughput. This service should be used by traffic streams with critical delay requirements, such as conversational applications. A second portion should be allocated to a service that can give less stringent guarantees on delay but still offer a high throughput. This service is offered to traffic types that have less stringent demands on delay. These two portions should fill up a certain percentage (50-90%) of the average available resources. It is important that the system is not overloaded by

[3]Adjacent antennae, base stations, etc., simultaneously transmitting on the same frequency, may be regarded as neighboring bins.

the services that have requirements on delay and throughput, since it would become impossible to fulfill them. To make the system really efficient and enable the scheduler to take advantage of the variations in channel quality, we introduce a third service class, a "stuffing" or "best effort" class, that we can use for traffic without any specific requirements on throughput or delay, to fill up the 10-50% gaps we on average introduce by not overloading the system with strict service-demanding traffic.

Traffic using the "best effort" service should neither suffer severe delays nor low throughput as long as it is admitted into the system. But, if congestion or saturation occurs, this service class will be the first to come in question for dropping a connection, since it is assumed to generate the least revenue per utilized resource.

1.4 Scheduling and Admission Control

A fast short-term resource scheduler should not need to handle *all* the incoming service requests. The scheduler will be unable to handle long-term variations in the resource demand, leading to buffer overflow in the case of an over-loaded system. The scheduler is only capable of serving the offered traffic by distributing the *available* resources. It can cope with short-term discrepancies in the supply-demand interaction, but if the long-term average load is larger than the total available resources, the scheduler will eventually fail. Therefore, an admission control mechanism should maintain an appropriate load on the system, relieving the scheduler from that task.

The admission control algorithm should arbitrate which clients that enter into a particular scheduler's domain. It should also select which clients should be removed from a scheduler's domain when the work load becomes overwhelming, either by dropping a client or by transferring it to another scheduler's domain. Admission control also sets the service level limits for a client, by assigning upper and lower limits for the throughput, a target delay, and a target error rate.

The admission control may consult the price models in the service level agreements, in order to make cost efficient decisions.

1.4.1 Common Objectives for Link Schedulers

In the design of a link scheduler there are mainly two directions for service provision that are not always easily combined. These two directions are the achievement of *Quality of Service* (QoS) and the achievement of *fairness*. Achievement of QoS requires that a certain *absolute* level of service is given to the involved data streams, whereas fairness requires that the flows receive a (weighted) *portion* of the available resources or capacity.

Both fairness and QoS can often be provided on wireline networks with predictable service levels and resource costs. In the wireless world, the available capacity is not easily predicted, and it is important to utilize the resources efficiently. In our view, there is no sharp distinction between fairness and QoS, since they reflect only different degrees of flexibility in the QoS demands and the clients' willingness to pay for the services. Therefore, we choose not to use any of these terms, unless necessary, in the continuation of the thesis. Instead we will discuss *service level control*, which in some cases or aspects resembles QoS and in other resembles fairness, and let our schedulers and admission control work toward maximizing revenue for the network operator.

Predictable Service Provisioning over Wireless

Receiving a pre-defined service level is not only attractive, but also necessary for certain applications. Service level control can, and should, be provided also for wireless links, as long as the capacity suffices. However, it is necessary to allow some flexibility in the definition of the service levels. The service quality must in some cases be allowed to adapt to the circumstances, since the resource quality and availability varies. In an extreme situation the channel capacity may vanish, so even allocating all resources to one client, will not help. This is an extreme situation, and in most cases, the scheduler should be able to exploit the variability in a positive sense, hiding the variability from the clients.

In the next chapter we will elaborate on this, including pricing models and Service Level Agreements, that reflect a customer's willingness to uphold a service level, and the compensation related to not receiveing the agreed service level. This framework also incorporates fairness issues, but implicitly, as the scheduler's objective to manage the load offered by the admission control in a cost efficient way.

1.4.2 A Framework for Service Level Control over Wireless Networks

To summarize before we move to the next chapter, where we will discuss the resource allocation from a business point of view, we are aiming at creating a framework that will provide a link between the revenue and the scheduling performance:

- Resource efficient (spectrally efficient) scheduling, handling the traffic provided by
- revenue maximizing admission control, based on the revenue models from
- Service Level Agreements, signed between the network operator and the service providers, reflecting the value of the needs of the end users.

Chapter 2

Revenue - the Criterion

In this chapter we discuss business models for future wide-area covering wireless mobile networks. The approach in this work is as direct as possible: Maximize the gain for the stakeholders involved in the different phases of a wireless network life cycle.

The chapter begins with an outline of the involved stakeholders or actors followed by an outline of some existing and suggested business models. The business models describe how the actors may interact in order to generate revenue, and in some cases, to make an investment economically feasible and therefore possible at all. We then look at the interactions during three phases in the wireless network life cycle, namely: The deployment phase, the operation phase, and the exit phase. The main focus will naturally be on the operation phase, where the contribution from a signal processing point of view will be most obvious. But this view is also important in the deployment phase, since the network planning takes place in this phase, and we have the possibility to choose designs that will enable an efficient operation phase. The operation phase occurs when the network is running in "steady state". Possible business and pricing models for this phase are then described and discussed.

A framework with Service Level Agreements (SLA) is outlined, and the conclusion is that in order to obtain as high a revenue as possible, the air interface must be as flexible as possible, maybe even spanning over several access technologies. A general resource manager, including admission control and short term resource scheduling, is required to direct traffic through the most efficient path.

We have chosen to look at revenue maximization as the criterion to use in the operation phase, when allocating resources to different clients. It may be argued that other criteria, such as user satisfaction, or resource utilization fairness, are possible. However, it is our working assumption that any reasonable criterion should be possible to map, through a pricing model in an SLA, to a revenue criterion.

2.1 Who Are the Actors?

A normal way to understand how companies are created to offer products, such as goods or services, is based on the insight that something is needed on the market, termed *market pull*. A company specialises in providing the required product at a certain market, since there is a gap to fill. An alternative way, that is commonly referred to in the scope of high-tech products, is the *technology push*, where it is said that the companies inventing the product (or other interested parties) are creating the market need by influencing the public opinion through various marketing activities.

Since there are several stakeholders involved in a high-tech infrastructural project, the case is more like a chain of *market pull* companies, with at least one *technology push* company somewhere in the chain, the latter often facing the major risk in the project. In the background, *regulators*, such as governments, overlook the development. They play an important role, since they say what goes and what not.

Example 2.1: Market Pull companies

Electronics components manufacturers and solid state electronics manufacturers do business as usual. They have to provide their components to the equipment manufacturers. They compete with other component manufacturers in order to be selected for the final equipment. They mainly act as "Market Pull" companies, trying to provide and integrate the required functionality in their components.

Example 2.2: Technology Push companies

Some equipment manufacturers act much as technology push companies. Take the integration of digital cameras into mobile telephony handsets as an example. It is not obvious to all end users that a digital camera is required together with a mobile phone. However, equipment manufacturers believe that the need can be created by means of marketing. The reason for integrating several technologies into one gadget is to take additional market shares from other market segments (in this case from the digital camera market), thereby increasing the amount of money that consumers are willing to pay for the equipment. At the same time, the network operators hope that end users will also pay for sending the digital images over the mobile network.

2.1.1 Deployment

In the deployment phase, the main actors economically involved are banks, governmental bodies, network operators, equipment manufacturers, building contractors, real estate owners, and venture capitalists. The number of users is small and the primary target for the service deployment is to achieve good coverage (rather than high capacity), so that the early users accept the new services offered as valuable and useful. In Figure 2.1 we illustrate how the different actors may interact during a network deployment. Banks are

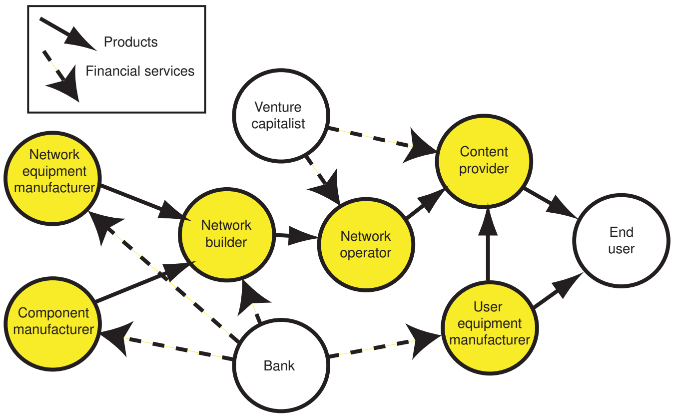


Figure 2.1: Example of how the business interactions may take place in a network deployment phase. We only show the product flows, including financial services, in this example. There is of course also a reverse flow of money.

not willing to take high risks, so they provide financial strength to stable companies far back in the value chain. Venture capitalists, on the other hand, are supposed to invest in higher risk ventures. They can be found where the expected payback is high, that is, near the top of the value chain, near the end user. Component and equipment manufacturers provide system components to system integrators, or network builders. The network builders deliver operable networks to the network operators. The network operators then sell transport services to the content providers (or service providers), that in turn sell their content to the end users over the network.

In the case of today's deployment of 3G networks, the companies that have been forced to take the blow when "the market" seemingly fails, have been the network operators. Furthermore, many of the operators have paid large amounts for obtaining the frequencies required for 3G operation [60]. Of course, their problems have propagated backwards in the chain, to venture capitalists, banks, and equipment manufacturers, who have to accommodate big debts from the operators, debts that may never be paid at all.

2.1.2 Operation

By applying more refined signal processing, an existing network can be enhanced to offer a higher service quality, improve capacity, be run at less cost, or even all at the same time. It is a matter of cost and value whether a more sophisticated method should be chosen to replace one existing. If the expected increase in revenue is higher than the expected cost, including risks, then the investment should be done.

During this phase, the idea is that the infrastructural system should be accessible to the users. The users, or rather - the customers, are now the source of revenue for the involved actors. They pay for services accessed through the network. As the number of users and services increase, network capacity has to increase.

2.1.3 Exit

A stakeholder should be able to exit from the venture at a certain point. Either, a stakeholder could sell its shares in a phase when the expected payback is yet to come, or he could transfer his interest to a new venture in a phase when payback is considered complete.

2.2 Business Models

A business model describes how a company does business and with whom. Especially it tells whether a company should *make or buy* key components for their products. Looking at a case of a service provider that wants to offer e.g. secure wireless access to corporate intranets, he could choose between producing or buying the wireless access to the users, and between producing or buying the secure access to the corporate networks.

Definition 2.1: *End user*

An *end user* is an actor that pays for using services transported over a wireless network. The end user may or may not in turn make revenue from his utilization of the services.

■

Definition 2.2: *Service provider*

A *service provider*, or equivalently, a *content provider*, is an actor that makes revenue from providing end user services over a wireless network.

■

Definition 2.3: *Network operator*

A *network operator*, or equivalently, a *network owner*, is an actor that makes revenue from providing network access and transport services over his wireless network.

■

In this outline, we make the distinction between service providers *making or buying* the wireless access to the end users or subscribers.

2.2.1 Single Service Provider

This is the traditional "monopoly" situation in the wireless telecommunication business. The network operator also provides the end-user services, such as voice telephony and Internet access.

2.2.2 Multiple Service Providers

In a different business model that could be used, the end user subscribes to a service provider, or a content provider, not to a network operator. The user wants to access a certain service he finds useful. In this case, the network is merely a bearer of the service, a way for the content to reach the user. Subscribing to a network operator will be an issue for the service or content provider, not for the end user. The service provider will choose to *buy* the wireless access service from the network operator.

Example 2.3: Subscribing to a Service Provider instead of a Network Operator

---

Say Microsoft has developed a wireless “Outlook” client for business use. Through monthly license fees, the customer company will buy this service from Microsoft, a service that includes access to both corporate and Microsoft-owned servers, “anywhere, anytime”. Then Microsoft “owns” the problem of achieving the wireless access, and buys the soultion from one or several wireless network operators. The different wireless network operators may use different pricing policies, making their services more or less attractive to use under different circumstances.

We have seen a development in other infrastructural areas, such as railways and telecommunication companies (telcos), towards this business model. In Sweden, for example, there was traditionally a monopoly situation in the railway business, where the same Government owned company (Statens Järnvägar) owned the rails and ran the trains. In 1988 it was split into two parts, one responsible for running the trains, and one responsible for maintaining the railways. Later, in 1995, the Swedish parliament decided that trains should be run in competition with other companies, thus sharing the same rails among different “transport service providers” [71]. This has been further expanded by making it possible to buy trips to places even without railways, through collaborations with coach companies and car rental companies. This resembles using different access technologies for the same service, in a telco context where the service is independent of the technology.

In the telco case, de-regulation in Sweden in the late 1990’s allowed new actors to compete with the previous sole operator Televerket (later Telia and TeliaSonera) using the same access network. However, Telia kept control of the access lines to the subscribers, making it necessary for a user to subscribe both to the access service (with Telia) and to the telephony service provider (with Telia or any competitor). This is fortunately slowly changing to the better as competitors are allowed to access the local Telia telephone stations with their own equipment.

The multiple service provider business model offers the best conditions for competition at the service level. Many service providers can get involved in the operation phase, generating a high expected revenue for the network operator, who in turn gets an incentive to maintain, enhance, and develop the network. In the case of the existing 3G networks operators, their required return on their investments may only be reached if a broader view on their service provisioning is adopted. A possible value chain model for this scenario is outlined in [59]. It is within this business model that the ideas presented in this thesis will find their best use.

Of course there should be more than one network operator to choose from for the service provider, and the possibility to switch network operators should be facilitated by the usage of a standardized access technology, just as trains can run on different companies' railways as long as the rail widths stay the same. Multiple network operators improve competition and thus pricing and network service offerings.

A special branch in the wireless access business with multiple service providers seems to be deploying quite fast:

Example 2.4: 4G networks

Many public places, such as restaurants, cafés, libraries, malls, etc, offer wireless hotspot access for their customers to access Internet and other specific services. These networks are often referred to as "4G" or "fourth generation" nomadic wireless networks. It is an interesting development that is taking place, not only involving access points belonging to different "operators" but also across different access technologies. See e.g. [1] for some references on business models for these networks.

However, investigation of the optimal combination and use of different access technologies is outside of the scope of this thesis.

2.2.3 Advantages of Having Multiple Service Providers on One Network

Why should there be multiple service providers using a single operator's network? Wouldn't it be more efficient to also let the network operator run the end-user services? Then he would control all resources and be more flexible in allocating them to different services. Won't there be a waste of capital by having more stakeholders involved, that all want to earn a profit from their involvement? There are several lines of arguments that point toward the desirability of a situation with multiple service providers.

Richer service selection It is not likely that a single service provider / network operator would produce all types of end user services since different services require different pricing policies and different customer support, thus making it cumbersome for a large corporation to introduce a new small revenue service. Small companies offering limited revenue services, along with large companies wirelessly extending their existing services, will enrich the selection of available services and therefore increase the total possible revenue for the network operators [3].

Competition Several service providers producing similar services give the end user the option to choose one or another, resulting in competition between service providers. Each service provider will feel pressure to improve its offered services, in order to keep customers and to get new ones. Services will thus improve and prices will also probably drop.

Cost sharing The network operator will only pay for building, operating, and maintaining the network. All other end user service related costs will be covered by the service providers or their end users. Moreover, the pricing policy utilized by the network operator towards the service provider allows for a variety of cost or risk sharing setups by dividing the fee into a fixed part and a service related part.

It may of course be argued that there are disadvantages with a multiple service provider situation.

Lack of capital to invest in a large infrastructural project may be a problem. Since the 3G networks that are built today are financed to a large extent by the previously amounted profits from 2G service provisioning *in combination* with network operation, it is not a natural step to take for companies that have succeeded in 2G, to build a network and not take part in the end user service provisioning. Other combinations of actors[1] will have to be formed, and it may be a difficult task to find investors for these new, and different, formations.

Waste of capital may arise when similar services are developed by different competing end user service providers. All actors will have to be profitable in order to continue their activities. Thus, the more actors involved, the more money will be required to flow from end users to the service providers.

[1] 2G and 3G service providers are of course not omitted from these actors.

2.3 Revenue from Operation

In this section we expound on the operative phase of the network lifetime. Furthermore, we focus on the "Multiple Service Providers" business model.

2.3.1 Service Differentiation

The transmission services offered by the network operator should be general in the sense that they should support any reasonable traffic type. Anything from on-demand reservation of broadband data connections, through real-time multimedia conversations, to short bursts of application data, should be possible to host on the network. However, some care must be taken to ensure that the expected costs never exceed the expected revenue from adding a new service provider to the existing population. At some point it will be necessary to further deploy the network in order to accomodate a new service.

Different voice service providers could buy their user access from the same network operator. They may have different target customers that require different service levels. One could aim at high-end users that need a reliable service with high speech quality, being willing to pay more than the other service providers' target customers, that expect less from the service and thus have a smaller budget for the voice service. This could also be true for a single service provider, since different network services could be bought for different profile customers. Since the two user groups belong to two different service categories that may be bought separately from the network operator, they do not compete for the same resources from the voice service provider's point of view. This differs from the case where a single service provider also runs the network.

2.3.2 Service Level Agreement

Depending on the type of service a service provider is running over the operator's network, different pricing criteria could be adopted. A network operator and a service provider must come to a Service Level Agreement (SLA)[2]. It mandates the required service quality that the network operator should provide, but it may also put a limit on the amount of resources that can be consumed by a service provider or service class. An agreement may include maximum and/or minimum limits on:

[2]The SLA is a central component in our framework for revenue maximization in wireless network resource allocation.

- Number of simultaneous users (globally and locally, and perhaps time-varying)
- Active connection throughput and delay
- Usage of available resources (for one, several, or all connections)
- Connection establishment latency
- Pricing for normal operation (within the limits) and exceptions
- Portion of time that the SLA should be fulfilled
- Penalties for not fulfilling the SLA

Fulfilling all requirements will gurantee a certain revenue for the network operator. Breaking the SLA should lead to economic compensation for the suffering party, and a penalty for the breaching party. In the most probable cases, the penalties should be included in the SLA itself, thereby avoiding expensive disputes and external arbitration. It is thus necessary to monitor and trace performance and important events in the wireless network in order to ensure that the SLA is fulfilled. In case of dissatisfied users or customers, it should be possible to deduct from the network traces and reports whether the SLA has been fulfilled or not. If the SLA was fulfilled, then the service provider should consider a re-negotiation of the SLA, in order to buy a better network service for its customers. If, on the other hand, the SLA was not fulfilled then the network operator should consider an upgrade of the network or a re-negotiation of the SLA.

Overbooking

An opportunity for the network operator to earn more money is by overbooking the resources. The operator then signs SLAs that he will most probably not be able to fulfill when the demand becomes high, e.g. at peak hours. At these events it is the task of the admission control to maximize the revenue for the operator, in the long term by not excessively breaking any SLAs, and in the short term by carefully choosing which SLAs to break. Admission control is further discussed and explained in Chapter 3. The scheduler will play an important role in minimizing the damage by efficiently allocating the available resources to the remaining clients. This is a calculated risk taken by the network operator in order to increase revenue at the expense of damaging the trust in the service. The theory of this behaviour is referred to as *yield management*. It is found in business areas where

- the resources cannot be stored for later use, and,
- the same resource can be sold at different prices to different customers at different times.

Examples of such resources are hotel nights, flight seats, and in the wireless communications case; channel resources. See for example [55] for an introduction to yield management.

A peculiarity with wireless communications when dealing with overbooking is that a wireless channel resource bin is a fixed resource amount, but with a varying service rate (channel capacity).

Definition 2.4: *Best effort service*

A *best effort service* is a transmission service without stringent service requirements. It is provided using resources available after having provisioned other service classes with more stringent service requirements.

■

Example 2.5: Best effort flight ticket analogy

A stand-by ticket for a flight can be seen as a best effort service. The traveller (the service user) has no requirements on exactly when to fly, or where to sit in the airplane. The traveller's only requirement is to eventually arrive at the destination. For the airline company (the network operator), the stand-by service is a means to compensate for the cost of otherwise flying with empty seats.

As in the flight analogy in Example 2.5, there must be a substantial difference in pricing between best effort and guaranteed services, also in wireless networks. A guaranteed service will in a wireless network require a certain capacity, for which the required amount of resources are not known in advance, whereas a best effort service user will accept what is left over. It may be understood that an appropriate mix of guaranteed and best effort services provide the best basis for a good revenue for the operator. We should avoid the risk of breaching many SLAs by carefully calculating the level of overbooking of guaranteed services that can be supported by the system. Moreover, the remaining variations between high and low usage from guaranteed-service customers, variations that depend also on user mobility and usage patterns, should be filled with best effort customers.

Example 2.6: Guaranteed and best effort service mix

In Figure 2.2 a randomly generated example shows how the total system capacity and the demand of guaranteed services (QoS[3] demand) may vary over time. It is desirable to fill up the gap between the QoS demand and the total system capacity with best effort traffic. Since it is expected that QoS traffic would generate more revenue per resource unit than best effort traffic, QoS should fill up a large portion of the traffic mix. However, we do not want to risk too high outage probability[4] for QoS customers, indicating that we should moderate the portion of guaranteed services.

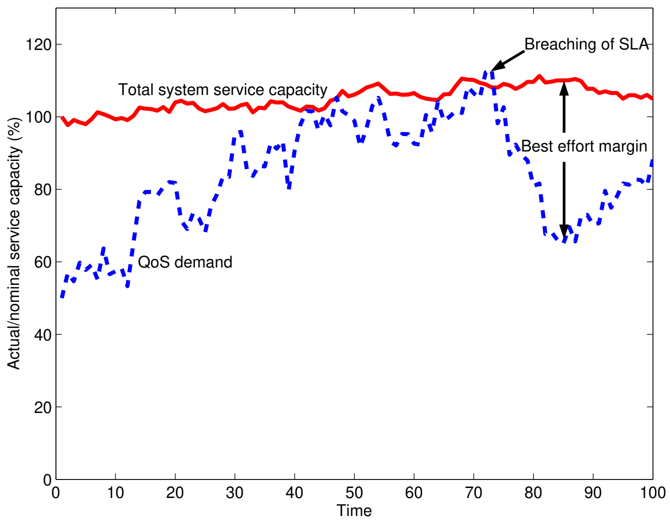


Figure 2.2: Example of a service mix and how the QoS demand may vary over time. A good mix of SLAs should maximize the expected profit, including the risk of breaching some SLAs and paying some penalty fees.

[3] "Quality-of-Service" refers to provisioning of services with predictable quality.
[4] The outage probability is the probability of not being able to serve a client.

Contingent Pricing

The penalty that the network operator has to pay to the customer according to the SLA could be regarded as a special case of *contingent pricing*. Then there is an agreemet between the seller and the buyer, that if a buyer is interested in booking a service at a low price, then the seller offers a compensation to the interested buyer, should the seller later find a different buyer offering a higher price for the booked service. Contingent (uncertain) pricing thereby helps both the seller and the buyer to reduce risks in a transaction. It also has the effect of prioritizing between customers that value the same resource differently. A customer that needs the service more badly will pay a higher price than another customer, and thus be a more profitable choice when running short of resources. Contingent pricing is explained and analyzed in [16].

2.3.3 Pricing Models

A simple model for pricing is to pay for the service that you get, completely proportional to the usage. This fits very well into a best effort service, where there are no explicit service level demands for the service to be meaningful. This simple case is illustrated in Figure 2.3(a). There is no penalty on low service provisioning, and no fixed fee to protect the network operator from low usage. Thus, the pricing model has no incentives for providing any service level guarantees.

However, when strict service level demands are introduced, this simple best effort case is not adequate. Some extended pricing models are defined below, that together with Figure 2.3(a) to Figure 2.3(f) serve as examples of what could be used when we need to take the service level into account.

1. Simple proportional pricing without strict service requirements
2. Fixed pricing for fulfilling the minimum SLA requirements and a penalty for not fulfilling them
3. Proportional pricing with a penalty for unfulfilled service requirements
4. Proportional pricing with a ceiling and a penalty for unfulfilled service requirements
5. Progressive pricing when running short of resources, in this case with a service ceiling

6. Progressive pricing when running short of resources, in this case with a price ceiling

The network operator must fulfill all the SLAs minimum requirements, otherwise he will suffer a penalty fee. This is a minimum level of service that the operator must maintain, and doing so will guarantee a certain revenue.

Note that we must distinguish between an *aggregate* pricing model and a *per-user* pricing model in the SLA. The aggregate pricing model should give the network operator an incentive to provide acceptable service to as many users as possible under the respective SLA, whereas the per-user pricing model should regulate what an acceptable service level is for an individual user. It should also allow for some limited service level flexibility in the case that users have extremely bad channel conditions and thus cost too much in terms of system resources to uphold.

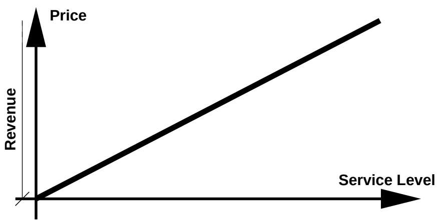
The pricing models presented above are applicable both to per-user and aggregate services. The difference is in the meaning of the "Service Level" axis.

- In the aggregate case, "Service Level" may represent the number or portion of the users under a certain SLA that receive a satisfactory per-user service.
- In the per-user case, "Service Level" may represent the average data rate, or the portion of the data delivered timely according to some delay contraints, or a combination of both.

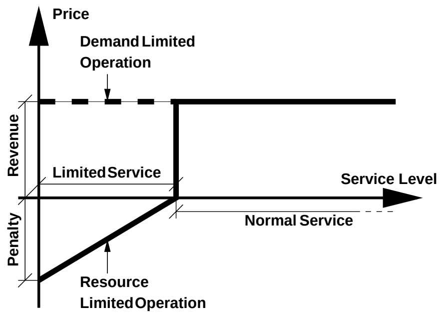
The pricing models may be applied either per-user, or on the aggregate service, or even both simultaneously, see Example 2.7. However, we stress that even though the pricing models can be applied per-user or on aggregate services, the price discussed is the one paid by a service provider to a network operator. What the end user pays for obtaining the service is an issue between the end user and the service provider, not in the scope for this thesis.

Example 2.7: Per-user and aggregate pricing simultaneously

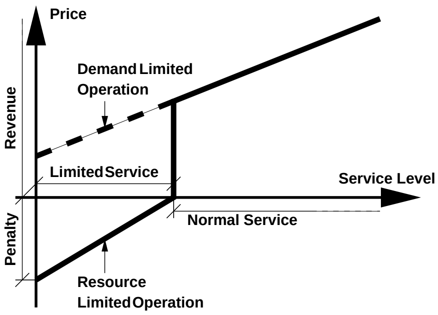
A network operator serves n users belonging to the SLA of service provider A. The fixed pricing model in Figure 2.3(b) is used for the per-user service, whereas the proportional pricing model with QoS in Figure 2.3(c) is used for the aggregate service.



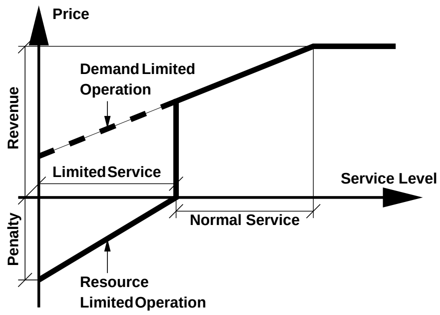
(a) Model 1



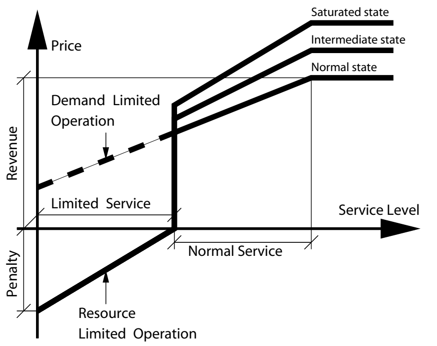
(b) Model 2



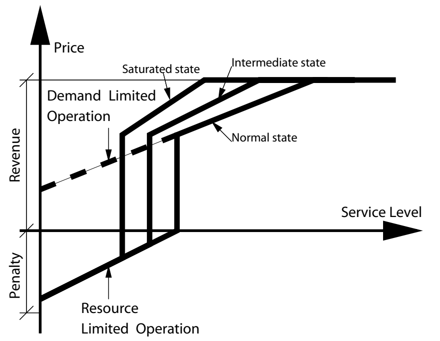
(c) Model 3



(d) Model 4



(e) Model 5



(f) Model 6

Figure 2.3: Six different price models applicable to wireless services. See page 25 for an explanation to the price models. The different states in models 5 and 6 are defined in Definition 3.2 in Chapter 3.

This means that the network operator will receive the per-user fee for serving each of the users, as long as their service level is above the threshold. In this case, “Service Level” in Figure 2.3(b) refers to a per-user service parameter, such as throughput or delay.

The network operator will also receive an additional aggregate service fee for serving $n \geq x_A$ users, where x_A is the “Service Level” threshold of SLA A in Figure 2.3(c). Thus “Service Level” in Figure 2.3(c) represents the portion of users that receive an acceptable service. However, if $n < x_A$, then this fee could be negative, should there be unattended users requiring service.

There are also requirements in the opposite direction: The customer has to pay a fee even if he is not utilizing all the services he is entitled to. This is illustrated by the dashed lines in the figures, where the system is under demand limited operation, meaning that the service demand is less than the service supply. In Example 2.7, this may be the case when $n < x_A$, but no unattended users require service. It is not the fault of the network operator that the service is under-utilized, so the network operator will still demand a fee. In Example 2.8 another case with a fixed pricing model is outlined.

Example 2.8: Fixed pricing under different usage levels

The fixed pricing model 2, also seen in Figure 2.3(b), allows the network operator to receive a fixed fee for the service, regardless of the usage. Under this pricing model there is no direct additional gain in fulfilling more than a minimal requirement. If the resources set the limit, so that the agreed minimum service level cannot be provided, then the network operator will have to pay a penalty to its customer. This minimal requirement is found in Figure 2.3(b), at the point where the dashed line changes into a continuous one. We can see that even if the customer obtains a higher service level, then the network operator will not increase its revenue.

It is in the operator’s interest to increase the revenue if the cost is not expected to increase more. Depending on what pricing models are applied, and depending on the extra demand from users currently not included by the minimum SLA requirements, actions can be taken to increase revenue.

Example 2.9: Increasing revenue

The proportional pricing model 3 allows a high flexibility in the normal service region, as seen in Figure 2.3(c). Under this model it could be fruitful to add another connection when the system state allows, since this will give the network operator additional revenue. Again, if the network resources saturate, then a penalty fee will be paid to the affected customer.

The outcome from the pricing policies becomes interesting when the system actually is saturated. There should be an incentive for the network operator to increase the capacity if this saturated state is reached frequently. One incentive is the penalty that he will have to pay to service providers with unsatisfied SLAs. A different way to look at the resource shortage is by the traditional supply-demand interaction. When there is a shortage in supply, prices tend to increase, whereas when the market is oversupplied, the prices decrease. In Example 2.10 these cases are illustrated.

Example 2.10: Progressive pricing policies

In Figure 2.3(e) and Figure 2.3(f) the two different progressive pricing models 5 and 6 are illustrated. Services become more expensive when the network operator runs short of resources[5]. This is illustrated by the transition from one price/service curve to another when the system becomes highly loaded in Figure 2.3(e).

However, if model 5 or 6 is used for some customers, and the service level has been broken for some other customer leading to a penalty being paid, then it may be fruitful to further break that SLA in order to accomodate more users from the class using pricing model 5 or 6. The extra income from customers under model 5 or 6 can be used to compensate the overlooked customers. To avoid this situation, it is important that the transition from "normal state" to "intermediate state" is made only on temporary high service demand peaks, rather than on a continuous shortage of resources.

In Figure 2.3(f) we illustrate another alternative for a progressive pricing model. In this case, the customer desires a ceiling on the price paid, and agrees to receive a lower service level at the same price when the system

[5]The exact definitions of the different resource availability states are given in Definition 3.2 in Chapter 3.

becomes highly utilized.

In [8] a model is presented for mapping radio resource allocations to revenue, through utility-based functions and user acceptance of price and performance. It is there concluded that radio resource management policies and pricing policies should be addressed jointly, in order to tune the performance of the system. In our approach, this takes place in the phase when SLAs are negotiated between the network operator and the service providers. The service providers need to keep in mind the utility and pricing acceptance from their end users, whereas the network operator must consider the resource cost for offering the network services requested by the service provider.

2.3.4 Business Models and Pricing Today

GSM operators today act as both network operators and service providers. There is an exception from this, and that is what in Sweden is callad a *virtual operator*. The virtual operator buys bundled capacity from one or several "real" operators that own a network, and sells its services to subscribers over the existing network. As it looks today, these virtual operators can offer the real operators predictable revenue, in return for surplus network capacity. The only *contents* that these virtual operators offer, is the same as that offered by the real operators: Speech and messaging services. Most of these virtual operators offer the contents at a lower price than the real operator. Some offer additional services in order to target other customers. An example of that is NewPhone in Sweden, that offers a subscription where the subscriber receives a new mobile phone regularly.

In [3], criteria for the success of Mobile Virtual Network Operators (MVNO) are given, along with an outline of possible business models for them. MVNOs are there divided into two main categories and four sub-categories:

- Completing knowledge and resources

  Industry oversteppers Existing actors with strong trademarks in other business areas that wish to bundle telecom services to their customers along with their existing services.

  Niche actors Existing niche actors that want to offer their customers a complete service.

- Competing knowledge and resources

Telecopies Want to offer the same services as the real operator. Often relies on regulation to be able to enter the market, since it is hard to show a win-win situation vis-a-vis the real operator.

Market expanders Financially strong existing actors in the telco business, on a different geographical market, that want to expand into the local market.

Based on this categorization, the authors of [3] conclude that all four subcategories of virtual operators can succeed on the market, given that they focus on their strengths. For example, the *telecopies* will have to compete with lower prices, whereas *market expanders* should rely on their trademarks, technical competence, and existing customers to expand into new markets.

3G began its roll-out on the Swedish market during 2003. The operator Hi3G was first on the market for 3G services. Unlike Tele2/Telia and Vodafone, the other 3G operators in Sweden, Hi3G did not have any existing customers on the local market. Since Hi3G needs to take large market shares in order to survive in the long term, they have started a one-sided price-war, offering free voice and video calls within their network, for a limited time (currently 12 months). However, the other 3G operators do not make such a big deal out of the new technology, since they have their customer base [6]. Instead, they follow two different strategies:

- Offer high-profile wireless broad-band Internet access to corporate customers.
- Extend their wireless portal services over 3G access technologies.

There are still no indications of virtual operators entering the 3G market offering subscription-like services. However, there are third party companies offering their services over the operators' wireless portals, as we exemplify below.

High-profile Internet access

The current pricing model for 3G data services is a combination of flat-rate and pay-as-you-go [84]. The amount of transmitted and received data is summed over a period of time (one month) and the customer is charged

[6]It is argued in [60] that the 3G operators with 2G licenses will delay the introduction of 3G technology (and also delay and reduce the payments for their licenses) in order to to take advantage of and make revenue from the full potential of their 2G networks. The same paper has also demonstrated that most of the winning 3G license auction bids were uneconomical and irrational.

for the total amount. If the usage is less than a threshold level, then the customer is charged a minimum fee, whereas if the threshold is surpassed, the customer will have to pay the minimum fee plus a per-megabyte fee. In Figure 2.4 we have outlined this pricing model. Note that the service level is only defined in terms of megabytes per month. It does not say anything about delays nor throughput rates.

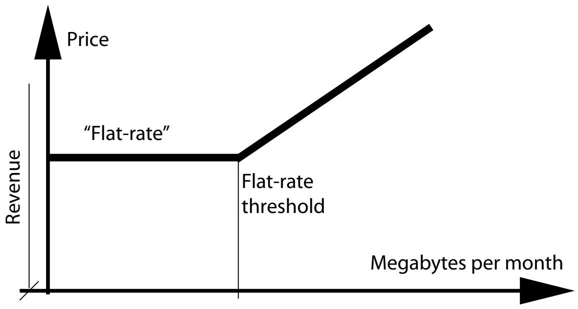


Figure 2.4: A common pricing model for today's 3G Internet access services. The minimum fee that the subscriber must pay to the network operator is given by the solid horizontal line. The service level is only defined by the amount of data transmitted during one month. Furthermore, the flat-rate fee coincides exactly with the price for the per-megabyte service at the flat-rate threshold.

Wireless portal services

All three Swedish GSM operators launched similar portal services almost simultaneously during the fall 2003. Through these portals, the end user can access operator-specific subscriber services, such as email or WAP, and some third-party services, such as mobile games, yellow pages, and location-specific weather forecasts. These portal services are also open for subscriber access from the 3G networks.

The pricing model here is a combination of free access to the portal and pay-per-view for the services. Again, there are no QoS guarantees included in the price model. However, users need a basic subscription to access the network.

An example of a similar strategy that has been successful is the i-mode service in Japan:

Example 2.11: i-mode in Japan

In the i-mode business model, users subscribe for a fixed monthly fee to the i-mode service through the NTT DoCoMo network operator. End users then pay an additional monthly fee for the usage of optional subscription-like services, supplied by third-party authorized content providers. The authorized content enjoys full integration into NTT DoCoMo's accounting and billing system, so that NTT DoCoMo handles the authentication, authorization, and accounting (AAA) and billing for the content providers, for a 9% billing commission. On top of this, NTT DoCoMo charges the user for the amount of data transmitted.

There is also a possibility for the i-mode users to access Internet content, reached through the web-browser in the handset. For the Internet content NTT DoCoMo will only charge for the data transmission. Content providers that choose this channel will have to handle their billing themselves.

From this success example we see that there is a potential business for third party content accessed through a mobile terminal. The business model for i-mode is similar, but not identical, to the *multiple service provider* model that we propose. The main difference is in the required subscription to the network operator, that we rather not have, since it ties the end users (and consequently the content providers), to utilize the network services provided by that particular network operator. Competition in both the wireless access level, and in the content provision level, is less efficient with an access subscription, than when a user can choose the content freely.

2.4 Discussion

We believe that the multiple service provider business model described in Section 2.2.2, along with the different pricing models and service differentiation, will improve the situation for both end users, service providers and network operators, as compared to the systems of today. We gave some motivations in Section 2.2.3, but there are, of course, more arguments, both in support of the new business model, as well as against.

What is required to make the business models suggested here feasible in future wireless mobile communications systems? The technology development and standardization should be aimed at creating a fundament for the introduction of service level agreements between network operators and service providers. They should come to agreements that they expect will be profitable for them.

The physical layer of such a technology should allow for a high flexibility in terms of resource usage policies that, among other things, enable fast allocation of channel resources where they are best utilized. Network operators could compete in this sense; being more flexible and efficient than the other, thereby offering cheaper access to the end users, and maybe even offering QoS guarantees.

In Figure 2.5 we see how the physical layer can be related to the service level and, through the SLA pricing model, result in a price tag for the offered service. The physical layer is here completely isolated from the pricing model, from the customer point of view. The customer only sees the service level and its price tag. The network operator, on the other hand, sees the cost for maintaining a certain service level for a user and thus has to consider what service level is the most profitable to offer.

The admission control, outlined in Chapter 3, has to keep track of what the total system resources are, and whether it can admit a new user without breaching other SLAs. A denied admission will result in a cost for the network operator, unless it already provides the maximum service required. An efficient scheduler, which is the main topic for this thesis, helps by utilizing the existing resources better. This is clearly seen in Example 2.12 using Figure 2.5:

Example 2.12: Service level, resource cost, and revenue

In this example we have a user belonging to a service class with a maximum and a minimum required throughput. This is seen in the price curve in the upper left part of Figure 2.5, that corresponds to price model 4 on page 25. We have outlined three of many different possible service levels[7] (A, B, and C), their respective cost in term of resources, and their resulting marginal profit for the operator. The user's average allocated channel conditions permits him to have an average allocated bin capacity[8] of about $2,6$ bits/symbol, as seen on the right horizontal axis.

We can see that, given a user with average allocated bin capacity marked

[7] The different service levels are results of different admission control strategies, which we will elaborate more on in Chapter 3.

[8] See Definition 4.3 on page 4.3.

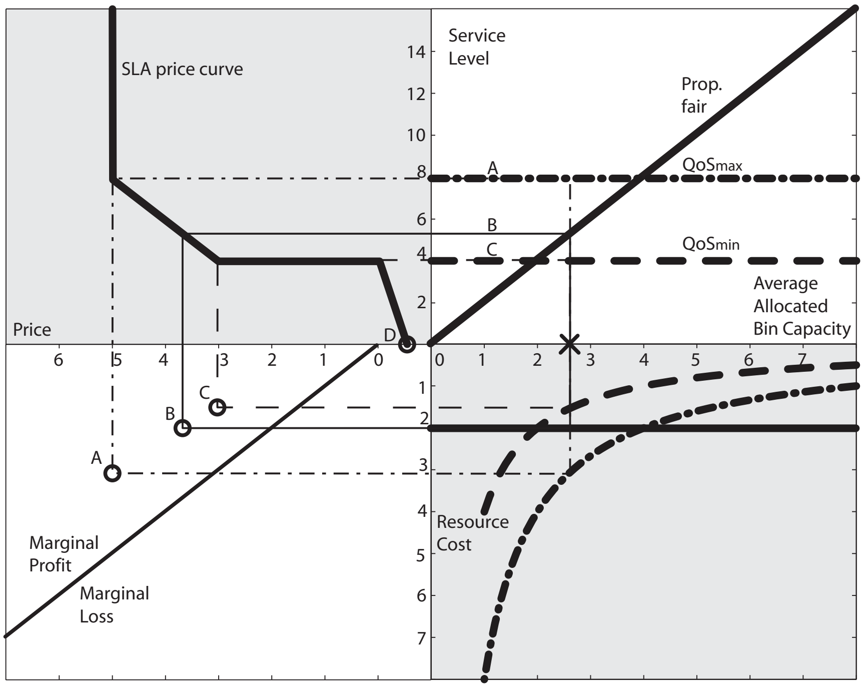


Figure 2.5: How the physical layer could be coupled to the pricing model and the resulting income. In the top left part, we see the price model curve. The top right part maps the average allocated bin capacity (see Definition 4.3) to the required service level (or vice versa). Depending on what service level is required, different amount of resources will be required to uphold that service level. We have exemplified with three different service level strategies: QoSmax, QoSmin, and Proportional Fair. *QoSmax* will always give the highest required service level, regardless of the cost. *QoSmin* will always give the lowest required service level. *Proportional Fair* will adapt the service level to the current capacity, so that the average resource consumption is kept constant for each user. The different service level functions are closely related to the resource cost functions, seen in the lower right part. Each service level function corresponds to a resource cost function, which is illustrated by the corresponding line styles. Thus we see that QoSmin has a lower resource cost than QoSmax for the given capacity. The resulting profit can be read out from the lower left part of the figure. The line dividing this part into a “Marginal Profit” and a “Marginal Loss” part represents the points where the income (the price) equals the resource cost. However, drawing this line requires knowledge of the actual resource cost, which may not be easily calculated in absolute terms.

by an “x” in the right horizontal axis, service level A results in the largest marginal profit for the network operator. We can also see that service level C results in the highest revenue/resource ratio, making it more attractive when the system is heavily loaded. There is a fourth possibility for the network operator, and that is to not admit the user at all, which results in the circle labeled “D”, near the origin. This service price is negative, which means that the network operator has to pay a penalty fee to the customer.

A more efficient scheduler would move the “x” on the right horizontal axis more to the right, since it would increase the average allocated bin capacity by better utilization of temporarily good channel conditions. This would result in less cost for maintaining the same service level, thereby resulting in higher profit for the network operator.

In Chapter 3 we describe how the admission control interacts with both the SLAs, and with the limitations imposed by the physical reality of the fading channels, in order to maximize revenue for the operator. In Chapters 4, 6, and 7 we demonstrate how the scheduling maps the available resources to admitted clients, in order to meet the service requirements. Finally, in Chapter 9, we will get the whole picture by applying the SLAs, the admission control, and the scheduling, on a (realistically) simulated system.

Chapter 3

Admission Control

The purpose of admission control is to serve as a gatekeeper to a server, admitting no more clients than the server can handle while offering an attractive service to the admitted clients. If too many clients are admitted at some point, then the service perceived by them may not meet their expectations.

Circuit switched communication systems, such as the GSM system, may use admission control to admit new calls up to a certain threshold, within the coverage area of a base station. The threshold is below the total number of calls that the base station can handle, and the marginal channels between the threshold and the maximum number of calls are called guard channels. The guard channels are kept to take care of handovers from neighboring cells, to allow higher priority calls, such as emergency calls, or, to maintain a good-enough quality for the existing users in an interference limited situation [11]. At some point, all the existing channels may be utilized. In that case, if a high priority call needs to be set up, then admission control can terminate an ongoing call in order to free resources for the high priority call.

In a Wide-band Code Division Multiple Access (WCDMA) system, which uses an interference limited radio access method, the threshold is dynamic, depending on the additional interference that admission of a new user would cause (see [50] and the references therein).

In multiuser packet-switched scheduled communication systems, such as the ones we focus on in this thesis, admission control is required in order to adjust the load on the link (the server) and its scheduler. Admission control here chooses which users, or clients, to serve, and their service levels, on a longer time-scale than that of the scheduler, in order to maximize the revenue in the light of a Service Level Agreement (SLA), as introduced in Chapter 2.

3.1 Overview and Notation

We choose to divide admission control into two parts that work on two different time-scales:

- Admission Policing (Session, or, call duration time-scale, e.g. 1-1000 sec)
- Service Level Control (Slow fading time-scale, e.g. 10-1000 msec. Depends on environment and mobility.)

To give the total picture, we also place the scheduler and its time-scale into this perspective:

- Scheduling (Small-scale fading time-scale, e.g. 0.1 - 10 msec. Depends on environment and mobile velocity.)

The purpose of this subdivision is to allow for some flexibility for the admitted sessions, in order to exploit, accomodate, or counteract the service rate variations imposed by the fading radio channels.

Admission Policing (AP) decides whether or not a new session, or call, should be admitted into the network. It also sets the limits for an acceptable service level that must be upheld by the scheduled link. This is further discussed in Section 3.2.

Service Level Control (SLC) adjusts the load on the scheduled link within the limitations given by the AP, reacting to the longer term variations in the fading channels, such as path loss and shadow fading. Service level control is further discussed in Section 3.3.

In Figure 3.1, a block diagram describes how AP and SLC interact with the scheduler, and how the transmitted data relates to admission control and scheduling. SLC states are defined in Definition 3.2 and the token bucket view was discussed in Remark 1.1 and defined in Definition 1.1.

As we pointed out in Remark 1.1 on page 4, we express service units in terms of *tokens*. This is an abstraction we choose to make since it allows for a general approach to the admission control and scheduling problem. Instead of specifying a data packet size, and then translating it into a number of resource units, such as a number of symbols, required to transmit that packet, we use the term token. *A token represents a fixed amount of data*

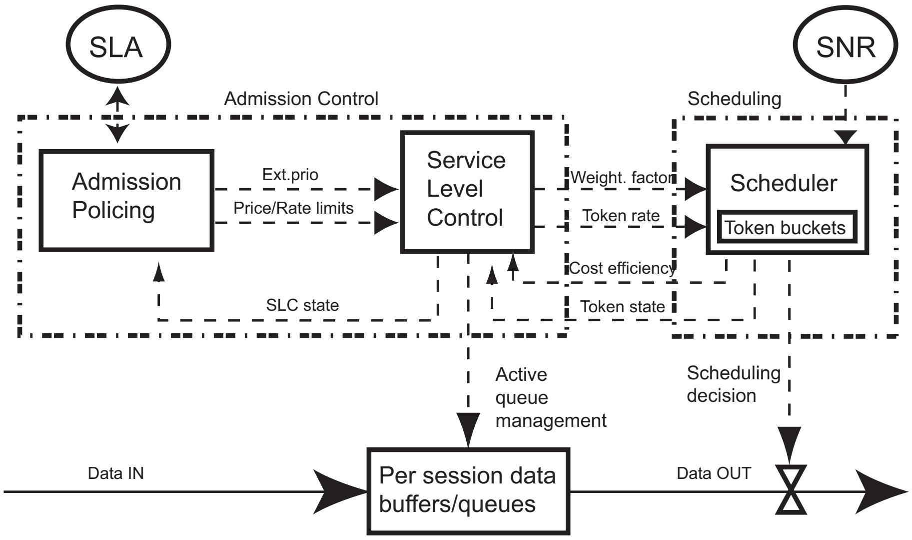


Figure 3.1: How the admission control is related to the scheduler. The scheduler receives the weighting factor from the service level controller, and returns estimates on the resource cost for serving different streams. The service level controller monitors the token bucket levels, and reacts to deviating values, by adjusting the token rates and the weighting factors. In this view, the client SLAs and the channel SNRs are regarded as externally imposed constraints. Active queue management may be used to drop or mark selected packets as a consequence of changing the service rates, in order to notify the communicating hosts of the changes. The SLC is the natural placement of such a functionality.

that can be served by one radio resource unit, when using uncoded BPSK modulation. Furthermore, the token abstraction allows us to disregard the burstiness of the incoming data, and instead present the resource scheduler with smoothly varying token bucket levels. When a resource bin is scheduled to serve a certain client according to the tokens representing its service demands, the data buffer for that client is drained with the correponding amount of data. Should the data buffer be empty at the moment of transmission [1], then the erroneously scheduled bin has to be marked as invalid for the scheduled client, and perhaps instead carry alternative information.

[1]There is a delay between the making of a scheduling decision and the actual transmission of the scheduled data.

Definition 3.1: *Admitted service rates*

An admitted service rate is defined by a *maximum* and a *minimum* rate limit. The rate limits are expressed in terms of *token rates* (see also Definition 1.1). The maximum (minimum) limit expresses the maximum (minimum) number of tokens given to a session in each scheduling interval.

■

The rate limits in Definition 3.1 are provided for the SLC from the SLA database, via the AP.

Definition 3.2: *SLC states*

Depending on the current load on the links managed by the the SLC, the SLC may be in one of three states:
Normal operation state means that at least one of the admitted clients is served at its maximum admitted service rate.
Intermediate operation state means that at least one of the clients is served at a rate higher than its minimum admitted rate, but none at its maximum rate.
Saturated operation state means that all admitted clients are served at, or below, their minimum admitted service rate.

■

Depending on which state the service level controller is in, different pricing models may be utilized. As presented in Section 2.3.3, the progressive pricing models 5 and 6 in Figures 2.3(e) and 2.3(f), respectively, may introduce different price labels according to the current resource availability. This enables for eager clients to increase their spendings in order to maintain a service level, if they employ price model 5, whereas clients using price model 6 will not tolerate higher prices and thus accept reduced service levels.

3.2 Admission Policing

The admission controller has an Admission Policing (AP) entity which is a logical unit that handles admission of new sessions, hand-over between cells and schedulers, and dropping of sessions in times of scarce resources, in a network-wide context. The admission policing has access to a database of all the Service Level Agreements (SLA) and long-term statistics on how they have been met. Based on this background information, it should apply its admission policy on new and existing sessions in the network, in order to maximize the revenue for the network operator. The AP sets the service level limits[2] for the service level controller, which works on a faster time scale, described in Section 3.3.

The AP also provides a measure of the revenue that is related to serving the admitted session at different levels. Should it be impossible for the SLC to comply with the given directives, the AP should react and e.g. adjust the limits, force a hand-over, or drop a session. The AP, together with the SLC, is thus a central component in our framework for revenue maximization in resource allocation, since it will govern which clients the operator will serve, and also at what rates they will be served, all in order to maximize the revenue for the operator.

By Example 3.1 to Example 3.3 below, we want to give a qualitative intuition to how an admission policy should operate, given the three SLC states above. A more quantitative explanation is given in the implementation example, in Section 3.2.1.

Example 3.1: Admission Policing in Normal SLC state

At a wireless access point let U active clients have ongoing sessions. At least one of the clients is utilizing the services at his admitted maximum rate. The access point is therefore not saturated and the AP may accept a new session from any client within reach.

As long as the SLC remains in the Normal state, there is no need to discriminate between clients belonging to different SLAs. In the other extreme, we have the saturated state.

Example 3.2: Admission Policing in Saturated SLC state

At a wireless access point let U active clients have ongoing sessions. All the clients are utilizing the services at, or below, their admitted minimum rate, and the SLC is therefore unable to adapt the service rates to the available resources. The access point is therefore saturated and the AP should not accept any new sessions. Actually, the AP should drop one or several sessions in order to take the SLC out of the saturated state. The sessions to drop should be selected based on their SLA:s and the clients' resource utilization

[2]Some applications do not accept any flexibility, while others may adjust to the current service level.

efficiencies.

Between the two extremes, we have the intermediate state.

Example 3.3: Admission Policing in Intermediate SLC state

At a wireless access point let U active clients have ongoing sessions. None of the clients is utilizing the services at their admitted maximum rate, but at least one is not utilizing them at its admitted minimum rate. The access point is therefore almost saturated and the AP should consider the possibility to transfer clients to neighboring resources, such as making hand-overs to different access points, or adjusting the limits within which the SLC may operate.

The aim in the intermediate state should be to avoid driving the SLC to its saturated state, since that would affect the performance of all the active sessions. However, it may in some cases be economically advantageous to admit high-revenue clients, despite the risk of driving the SLC into the saturated state. We provide a more detailed study of an admission policing entity and its implementation in Section 3.2.1.

In this outline, we have not made any distinction between *rejecting* a new session or client, and *dropping* an ongoing session or existing client. This distinction may of course be introduced into the SLA and the pricing models of Chapter 2. It is common to regard dropping as more annoying to a client, than rejection, since otherwise, the distinction would be meaningless[3]. In Example 3.1 we have therefore not particularly considered the case of a new admission driving the SLC into the saturated state, since this case will be handled, once in the saturated state. If we would make a distinction between dropping and rejection, we would not let a new admission drive the SLC into the saturated state without considering the consequences.

[3]If a rejection would be equally annoying or more annoying than a dropping, then never reject any clients and follow up by dropping the least profitable ones.

3.2.1 Implementation of an Admission Policy

Given the SLC states defined in Definition 3.2 and the corresponding admission policies presented in Examples 3.1, 3.2, and 3.3, we may now outline a detailed *example* implementation of an admission policing entity.

The required building blocks for an AP are:

1. A connection to a database containing data on the existence and current fulfillment of all SLAs,

2. a mechanism for managing handover of clients between neighboring servers,

3. a means for the SLC to communicate its state to the AP, and,

4. a means for clients requesting admission[4] to communicate their SLA membership and average downlink[5] channel quality[6], to the AP.

There are three states that the SLC can be in, as presented in Definition 3.2. In each state, three events can take place:

1. A client may leave the server,

2. a new client may arrive at the server requesting service, and

3. the SLC state may change.

In order to reduce the cases that the AP has to consider, we immediately decide that in the normal state, all admission requests are accepted, since in that state, the resources suffice and there is no need to discriminate between clients. Furthermore, the event of a client leaving the server, as in event 1 above, will not yield any actions from the AP, regardless of the SLC state, since that will free previously occupied resources. The four remaining cases, and the corresponding actions to take, are presented in Table 3.1 below. We describe the decision rules of Table 3.1 in Examples 3.4 and 3.5 below. In Table 3.1, e_{U+1} refers to the additional revenue (earning) generated by admitting client $U+1$ and c_{U+1} to the cost of rejecting $U+1$, whereas c_i refers to the corresponding cost for reducing the service level of client i. The cost of doing a handoff of client i to a neighboring cell is denoted h_i.

It is important to bear in mind that the costs c_i do not only include losing some revenue by reducing the service level, or the penalty fee for eventually

[4] Requests may come both wirelessly from mobiles and from the fixed network.
[5] We are primarily looking at resource allocation for downlink communications.
[6] The channel quality is estimated and provided over a random access uplink channel.

	Intermediate state	Saturated state
Arrival of $U+1$	Admit if $\sum_i^U c_i - e_{U+1} < c_{U+1}$	Reject
State change	Handoff $u = \arg\min_i h_i$	Drop $u = \arg\min_i c_i$

Table 3.1: Actions to take by the admission policing entity in the *intermediate* and *saturated* states, in the event of a client arrival or a state change. It is assumed that the state change event brings the state, into the one in the table, from the less critical state, i.e. from the normal state into the intermediate state, and from the intermediate state into the saturated state. The dropping in the saturated state should of course be avoided if it is possible to handoff clients to a neighboring cell instead. The action plan is quite simple. The difficulty is in calculating the c_i and h_i for all $i = 1, ..., U$, and e_{U+1}.

dropping an *individual* client i. It may also include a larger fee for breaching an SLA, by not fulfilling the *aggregate* network service agreement toward a content service provider. This is why we need the AP to communicate with the SLA database, in order to find which clients that can be dropped, without risking high penalties.

Having the decision rules indicated by Table 3.1, the remaining difficulty is to calculate the mentioned costs and revenue.

Example 3.4: Admission in the intermediate state

A new client $U+1$ requests admission while the SLC is in the intermediate state. The new client provides the AP with a measure of its average bin capacity, $\bar{C}_{U+1}$, as in Figure 2.5 on page 35. Based on $\bar{C}_{U+1}$ and the price model for client $U+1$ from the SLA database, the AP calculates e_{U+1} and c_{U+1}. An estimation of $\sum_i^U c_i$ is calculated using the corresponding price models and average bin capacities $\bar{C}_i$. If the cost for admitting is less than the cost for rejecting, that is if

$$\sum_i^U c_i - e_{U+1} < c_{U+1},$$

then client $U+1$ is admitted at its minimum acceptable rate. It is then the task for the SLC and the scheduler to make room for the new client, with help from the AP, since if the SLC remains in the intermediate state, then the AP will try to handoff some clients to neighboring cells.

Example 3.5: Admission in the saturated state

All admission requests are rejected while the SLC is in the saturated state. Furthermore, the AP will try to primarily handoff, and secondarily drop, clients until it drives the SLC into the intermediate state. It is desirable that, apart from consulting the SLA database as in Example 3.4, the dropped clients are the ones that represent the least revenue/resource ratio for the system, thereby releasing considerable resources (see Example 2.12 and the lower left part of Figure 2.5 in Chapter 2). If the SLC is successfully driven into the intermediate state, then previously rejected clients may be admitted.

3.3 Service Level Control

Based on the introduced service limits and revenue measures determined by the AP entity, the service level controller shall adjust the load on the scheduler. The service level controller must work within the limits assigned by the AP, as long as this is possible. The performance of the service level controller is monitored by the AP entity, that reacts to e.g. saturated service levels.

According to the revenue measures provided by the admission policy, the service level controller should take actions that are positive in the revenue generation sense. An example control strategy that a service level controller can use, is: When there is room for increased service levels, increase the service level of the most cost efficient streams; and when there is need for service level reduction, reduce the service level for the least cost efficient streams. Note that the service level controller works within the limits set by the admission policy, so there should be no risk that streams totally outcompete one-another: They will always receive at least the minimum service level (as long as it is feasible).

Furthermore, the different clients' service levels will have to be individually controlled in order to exploit the average bin capacity variations.

In Figure 3.2 we have outlined the general SLC problem.

The amount of tokens $r(t)$ in the bucket is a result of

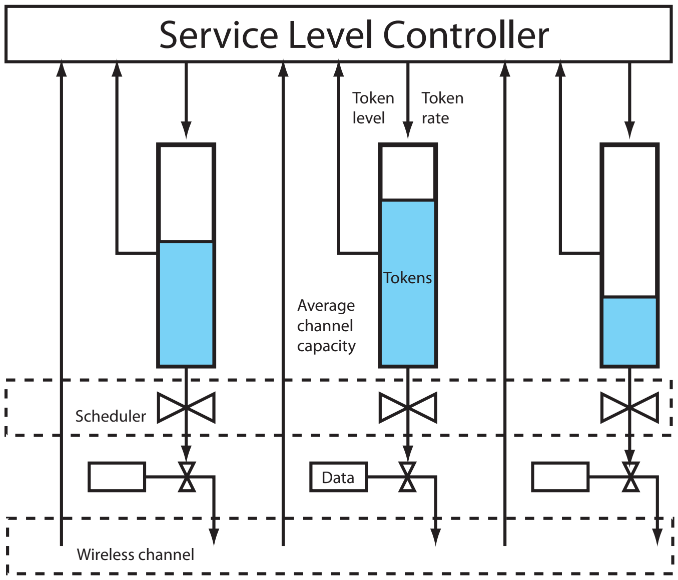


Figure 3.2: The service level controller, here depicted as a general controller, with the token levels and the average bin capacities as input signals. The outputs from the SLC are the admitted token rates, limited by the limits set by the admission policy. The SLC uses the input information in order to make an economically efficient decision for the service rates that will prevail until the next decision instant. The token rates are used to fill up the token buckets, until the next instant the SLC updates the token rates. For completeness, we have also included the scheduler and the wireless channel.

- the amount of tokens in the previous scheduling interval $r(t-1)$,
- the input rate $I(t)$; dictated by the data transmission service, shaped through admission control, and
- the output rate $a(t-1)$; dictated by the scheduler decision and the channel conditions in the previous scheduling interval.

This is represented by Equation (3.1):

$$r(t) = r(t-1) + I(t) - a(t-1) \tag{3.1}$$

The rate at which the token buckets are filled by the SLC should reflect the rate at which the data buffers are drained, since they consume the tokens by transmitting data. If the rate into the token buckets are on average higher than the rate at which the tokens are consumed, the token buckets will be unstable.

3.3.1 Service Level Control as a Linear Control Problem

The service level control (SLC) can be formulated as a linear, multiple input, multiple output (MIMO) control problem with constraints on the control signals. However, simpler controllers, such as PID controllers may perform well enough. For example, SLC could be handled by a single input, single output, (SISO) PID[7] controller, defined in Definition 3.3, with a slight modification yielding a single input, multiple output, (SIMO) PID. In Example 3.6 and Figure 3.3, we outline a proposal for how this SIMO PID could be achieved.

Definition 3.3: *SISO PID equation and parameters*
The main equation for a PID controller is given by [7, 72]

$$\Delta I(t) = K\left[e_p(t) - e_p(t-1) + K_I e(t) - K_D(y_f(t) - 2y_f(t-1) + y_f(t-2))\right], \tag{3.2}$$

where

$$e(t) = q(t) - y(t), \tag{3.3}$$

$$e_p(t) = bq(t) - y(t), \tag{3.4}$$

and

$$y_f(t) = py_f(t-1) - (1-p)y(t). \tag{3.5}$$

Here, K, K_I, and K_D are non-negative gain parameters, that dictate how much the different parts[8] of the PID controller contribute to the final control increment $\Delta I(t)$. In equations (3.4) and (3.5) we have two additional parameters, b and p, respectively. They are preferably set to $b = 1$ and $p = 0.5$ to begin with, while tuning the other parameters (K, K_I, and K_D). In Equation (3.3), $y(t)$ is the current output signal, that in our case is the *maximum* buffer level that is reported back to the controller, and $q(t)$ is the reference buffer level, thus $e(t)$ is the current control error to which the PID controller should react. The new admitted rate is then calculated as

$$I(t) = [I(t-1) + \Delta I(t)]_{I_{min}}^{I_{max}} \tag{3.6}$$

[7]A Proportional Integrating and Differentiating (PID) controller is a second order linear system that filters input signals to yield control signals that stabilize a system, and make it follow a given reference, in the presence of measurement noise and system disturbances. [7]

[8]The P, I, and D-parts, that stand for Proportional, Integrating, and Differentiating, respectively

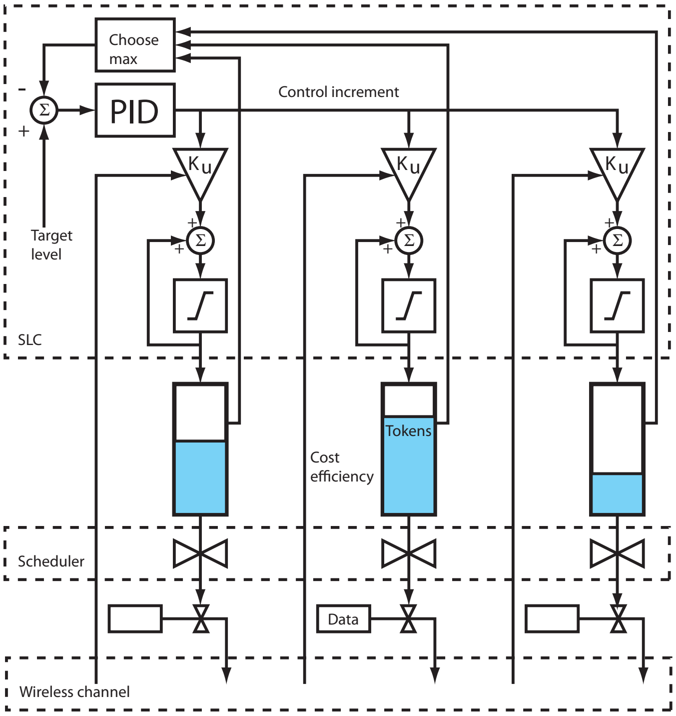


Figure 3.3: The scalar PID controller takes as input the difference between (a function of) the target bucket level (assigned by the AP, depending on the delay requirements and the admitted rates) and the maximum of (a function of) the actual bucket levels. The scalar PID thus has a single input signal. The output from the PID controller is an increment to the token rates, that gives an indication of whether they will increase or decrease, and also to what extent. The increment is fed through a client-dependent aaptive gain K_u, and a limiting function (the rate limits assigned by the AP), that together map the common increase/decrease signal to an individual increase/decrease signal for each client. The adaptive gains K_u are given by the clients' cost efficiencies, defined in Definition 3.4 below. As outlined in Example 3.6, a rate increase will then result in the most cost efficient clients' rates being increased most, and vice versa, unless the limiters become active.

where $I(t-1)$ is the admitted rate from the previous update instant, I_{max} and I_{min} are the rate limits as decided by the admission control and the SLA (see Figure 3.3), and

$$[x]_a^z = \max\left[\min\left(x, z\right), a\right].$$

■

The PID controller according to Equation (3.2) in Definition 3.3 is given in *incremental form*, since the output from the PID is the *increment* $\Delta I(t)$ of the control signal, and not the control signal $I(t)$ itself. An advantage of using incremental form is that it is possible to avoid *integrator windup* [58, 7] when any of the service levels is saturated, since we can use the true saturated signal $I(t-1)$ in (3.6) [72].

Definition 3.4: *Cost Efficiency*

Cost efficiency, E_u, is here defined as

$$E_u = \frac{\tilde{E}_u}{\frac{1}{U}\sum_{i=1}^{U}\tilde{E}_i}, \tag{3.7}$$

where

$$\tilde{E}_u = \frac{\Delta e_u}{\Delta I_u}\cdot C_u. \tag{3.8}$$

Here, Δe_u = (max price)-(min price) is the difference between the revenue (price) for the maximum and minimum transmission rate, ΔI_u = (max rate)-(min rate) is the difference between the maximum and minimum required transmission rate, according to the price model. Furthermore, C_u is the resource value for each channel (average allocated bin capacity) averaged over N previous time slots (see Definition 4.2 on page 58). The values in the vector $\tilde{E}_u$ (one for each client u) are divided by their average to normalized cost efficiencies E_u that average to one.

■

The purpose of defining the cost efficiency is that we will use it in the SLC in order to deduce which clients are better at utilizing their resources. This is important for the optimization of the admitted rates to the different clients, in order to utilize the available resources to generate the maximum possible revenue.

Example 3.6: SLC by a Single-Input Multiple-Output PID controller

---

A simple modification of the SISO PID in Definition 3.3 yields a multiple-output control signal, making it a SIMO controller. Having revenue maximization in mind, the modification could be carried out like this (use Figure 3.3 and Definitions 3.3 and 3.4 for reference throughout the example):

- The main idea is to modify the gains K in (3.2) for different users u (i.e. K_u), and also to let K_u have different values depending on whether $\Delta I(t)$ in (3.2) is negative or positive (rate decrement or rate increment).

- The scalar gain K is multiplied with a non-decreasing function of each client's relative resource cost efficiency (Definition 3.4), which is given by the channel properties ("Service cost" in Figure 3.1) and the SLA pricing information:
$$K_u = K \cdot f(E_u).$$
This will make the most cost efficient clients u increase their admitted data rates more than the others, by increasing their K_u.

- Moreover, if the scalar control increment from the SISO PID controller is negative, then we multiply K by the *inverse* of the relative resource cost efficiency:
$$K_u = K \cdot f(\frac{1}{E_u})$$
This will make the least cost efficient clients decrease their admitted data rates more than the others, by increasing their K_u.

All the resulting rate increments from (3.2) are added to the previous rates This approach will lead to a relative increase of the more cost efficient clients' rates.

Care must be taken in the tuning of the PID parameters in order to avoid oscillative behaviour, but also to make it fast enough to be able to track sudden changes in the resource utilizations.

Example 3.7: SLC by a MIMO controller

A Multiple-Input Multiple-Output (MIMO) controller is characterized by a MIMO transfer function from the input signals to the control signals. The control signals may be the individual token rates for each client, and the input signals may be the actual levels of each of the token buckets. The MIMO controller can exploit the dependencies between the different clients' token bucket levels. Additional inputs to the MIMO controller could be the channel properties for the different clients (and their reliabilities) and the SLA pricing information.

Generalized Predictive Control (GPC) [24, 25] is a method that promises near-optimal control strategies for such a MIMO approach. With GPC we can:

- control all bucket levels individually toward individual target levels, and
- take into consideration that the control signals are constrained.

However, we have not pursued this option. Instead we consider a non-dynamic approach for the SLC, as an alternative to which we can compare the PID controller, namely a linear program, described next.

3.3.2 Service Level Control as a Mathematical Programming Problem

A way of choosing the input rate into the token bucket is by means of a mathematical program, that maximizes a simple measure of the revenue generated by delivering the data.

The mathematical programming solution represents a different approach than that of the linear feedback control SLC. A linear program is able to maximize an objective function (that we could design to represent the revenue), based on a linear combination of a number of constrained variables and fixed parameters (that we choose to be the admitted rates, and their respective price tags). The constraints will be applied both to the admitted rates, and on the total available resources. By this approach, we hope to find revenue maximizing SLC decisions. We now describe in more detail how a linear program based SLC is accomplished.

Example 3.8: SLC by a Linear Program

Let us for now assume that we have a simple price model for all the served users: The users pay e_u money units for a certain data rate I_u, where $u = 1, ..., U$ is an index over the admitted users. They also only pay for data rates I_u betwen $I_{min} = \underline{I}$ and $I_{max} = \overline{I}$ tokens or data units per unit time. Let us now assume that the different users experience different average allocated bin capacities, namely C_u data units per resource unit[9].

[9]The average bin capacity is defined in Definition 4.2 on page 58.

In order to maximize revenue, the SLC should assign rates according to the following linear program:

$$E = \max_x \sum_{u=1}^{U} x_u \frac{\Delta e_u}{\Delta I_u} \quad (3.9)$$

subject to

$$\sum_{u=1}^{U} \frac{x_u}{C_u} \leq C \quad (3.10)$$

$$\underline{I} \leq x_u \leq \overline{I} \quad (3.11)$$

The maximized function E represents the possible revenue generated by serving the users according to the rate vector $I = [I_1, ..., I_U]$. In the constraint (3.10), C represents the total available resource pool (total number of resource bins). This linear program can be solved by standard methods to obtain the rates that optimize the current revenue.

For a two-user example, let us assume the following values,

$$\begin{aligned}
\underline{I} &= 50 \quad \text{[kbytes/s]} \\
\overline{I} &= 100 \quad \text{[kbytes/s]} \\
\\
\frac{\Delta e_1}{\Delta I_1} &= 2 \quad \text{[cents/(kbyte/s)]} \\
\frac{\Delta e_2}{\Delta I_2} &= 3 \quad \text{[cents/(kbyte/s)]} \\
\\
C_1 &= 3 \quad \text{[bits/symbol]} \\
C_2 &= 3 \quad \text{[bits/symbol]} \\
\\
C &= 50 \quad [(8 \cdot 1024)\text{symbol/s}],
\end{aligned} \quad (3.12)$$

resulting in the following linear program,

$$E = \max_x 2x_1 + 3x_2$$

subject to

$$\frac{x_1}{3} + \frac{x_2}{3} \leq 50$$

$$50 \leq x_u \leq 100$$

with solution

$$I = [x_1, x_2] = [50, 100].$$

From this example, we can make a number of interesting observations. They concern the solutions to the program. Changing the average allocated bin capacity for user 2, C_2, in (3.12) to have the value $C_2 = 2$ bits/symbol results in *any* feasible solution on the boundary (3.10), being optimal, since the average allocated bin capacities and the service prices cancel.

Changing the average allocated bin capacity for user 2, C_2, in (3.12) to have the value $C_2 = 1$ bit/symbol results in *no* feasible solution at all, unless users accept data rates outside their specified requirements.

Obviously, if a channel has average allocated capacity $C_u = 0$, then no resources should be wasted on it. In our problem formulation, we have the constraint (3.10) that will be impossible to uphold for zero bin capacities. In these cases, in order to keep the possibility to solve the service level optimization problem as formulated in (3.9-3.11), we have to omit users with zero bin capacity from the mathematical problem.

A different observation concerns the applicability to service level control as a dynamic system.

Observation 3.1: *A static solution to a dynamic problem*

The problem of assigning token rates that fill up buckets in order to control data delay and throughput, is a dynamic one. Linear programs do not have any memory to take previous decisions into account. A linear program will come to a solution that maximizes revenue, according to the price model (3.9) and (3.11), and given the snapshot of the system state, that we provide in the constraints (3.10).

■

We expect this to be an issue when applying LP methods to SLC problems, especially for two reasons:

1. The linear program will find solutions that assign maximum rate to good clients, minimum rate to bad clients, and an intermediate rate to one mediocre client. Thus the dynamics of a linear controller, such as a PID controller, will not be there implicitly to smooth out the rate variations. There are two sides to this feature:

   (a) The LP solutions quickly adapt to the current channel properties, which is good.

   (b) Some applications or communication protocols may have trouble with the possible fast rate variability.

2. In the linear program problem formulation, there is no explicit feedback from the token buckets, telling how many tokens remain from the previous round. This will have to be included, either as a post-correction to the linear program solution, or as an explicit constraint in the linear program, in order to control delays.

Above, items 1b and 2 will require special care when designing a system that uses linear programming for service level control. We will in Section 9.5 illustrate the effect of making a simple rate correction to the linear program solution, according to item 2. However, the possible problem of fast varying rates is not addressed in this thesis, but in the specification of services over such a system, the acceptable variability may be specified as a service level parameter.

3.4 Summary

Admission Control is here divided into two separate entities, Admission Policing (AP) and Service Level Control (SLC), each responsible for the system performance in their respective time-scales. AP utilizes the SLA database to admit the most profitable mix of clients, whereas SLC adjusts the service levels for the admitted clients. We have given explanations and provided examples for how this may be implemented in a communication system. In Chapter 9 we will see in more detail how these approaches affect the system performance.

Chapter 4

Scheduling

We will study a broad-band, down-link dominated, time-and-frequency-multiplexed wireless mobile system, capable of dynamically mixing different types of traffic over small-scale fading channels.

The preceding chapters have outlined the system from a revenue perspective. For the continuation of the thesis, the low-level details 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, on the service requirements.*

Adaptivity is crucial in order to obtain improved 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 approaches, such as coding and interleaving, perhaps with 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.

This chapter will serve as a background to motivate scheduling and to describe how it is carried out. In Section 4.1 we motivate the use of on-line fast resource allocation, or, scheduling. In Section 4.2 we describe the basics of scheduling, taking both channel conditions and service requirements into account. Finally, in Section 4.3, we discuss some practical aspects that need to be addressed when designing a wireless access network that utilizes scheduling, related to the acquisition of the information required for scheduling.

4.1 Motivations for the Use of Scheduling

The main motivation for the use of scheduling in wireless communications, is the combination of efficient utilization of the wireless channels, and the possibility for fine-grained adaptation of the resource utilization to the required service levels. Thus, it is a combination of *efficiency* and *flexibility* that motivates the use of scheduling.

We will in our approach utilize the flexibility for providing the service levels requested by the service level controller (SLC), from Section 3.3. By also improving the efficiency, we will be able to uphold higher service levels, or accomodate more clients, all with the intention of generating revenue for the network operator.

4.1.1 Motivation 1: Improving Spectrum Efficiency

The channel quality varies substantially over time and frequency, 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*, also called *shadow fading*, can be partly counteracted by controlling radio transmitter power. The remedy against *small-scale fading*, however, is traditionally different types of channel coding and interleaving.

As we outlined in the beginning of Chapter 3 (page 38), we recognize the different time-scales in the channel variations, and propose alternative methods for *exploiting* the variations instead of *counteracting* them. By measuring the time-varying channel quality, and using the measurements for link adaptation and fast multiuser scheduling, the available spectrum can potentially be used more efficiently, than by using a fixed resource allocation pattern and counteracting the channel variability by coding and interleaving.

4.1.2 Motivation 2: Fulfilling Service Requirements

Another issue for future data communication over wireless links, is that of providing stable and predictable data transmission services, despite the variability of the wireless channel.

The available bandwidth over the wireless channel is limited, so it has to be utilized efficiently. A way of doing so is to associate a value to the event of serving a certain client u with a certain resource bin s. This value is not trivial to find, since it should be based on both the service requirements for that client, and on the ability of the resource bin to provide the required service. Once it exists, it can be incorporated into the scheduling so that the scheduler can maximize the value of the transmission, and hopefully, provide the required service level.

In our approach, the SLC decides the service level that should be provided to each client. It is then the task of the scheduler to assign the proper resources on a short-term time scale in order to timely deliver the services to all clients.

4.1.3 Motivation 3: Channel Prediction Works

A central component in the scheduling approach is the mobile radio channel predictor. It has been demonstrated in [28, 74, 29, 30, 27, 73] that it is possible to predict the received power variations, quite accurately, for several milliseconds into the future, even for fast moving vehicular users.

That we at all require *predictions* of the channel conditions is mainly for two reasons:

- With predicted channel conditions, we can utilize the resources even more efficiently than outlined in Section 4.1.1, since we can plan transmissions over future resources, and thus increase the number of resources over which we optimize the schedule.
- The feedback loop *time delay* that is introduced by an on-line scheduling approach requires that we can predict the channel quality, far enough to be able to do the required scheduling calculations, but also to communicate the scheduling decision and prepare for its execution.

Having these predictions for all wireless channels, they can be used together with a target error rate to assign the feasible modulation complexity, as described in Section 4.2, for planning of the transmissions to the different mobile hosts. See also Appendix A.1 and references therein for a brief overview of modulation, coding, and link adaptation.

4.2 Optimization of Resource Allocation

Resource allocation deals with the problem of assigning shared server resources to clients performing 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, frequency, and space. The available spectrum can be represented in many dimensions, such as time, space/antenna resources, 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 resources that are allocated to different clients, can be partitioned by means of time multiplexing (TDMA), frequency multiplexing (FDMA), or even both simultaneously, into *resource bins*, that are the smallest radio resource transmission units dealt with by the scheduler.

The problem discussed in this chapter deals with methods for (sub)optimally allocating resource bins to clients. The allocation is based on the clients' requirements, represented by the amount of queueing tokens, and the clients' predicted wireless channel conditions, represented by their bin capacities, defined in Definition 4.1 below.

4.2.1 Channel Constraints

The channel SNRs are translated, via the target bit error rate (BER), to an allowed modulation format, as outlined in Appendix A.1. This translation results in an array of size $U \times S$, where U is the number of active clients, and S is the number of resource bins in the scheduling block. This matrix will be called the *capacity matrix* and is defined below. See also Figure 4.1, where an example with six clients and ten resource bins is outlined.

Definition 4.1: *Bin capacity $C(u, s)$ and capacity matrix C*

The *bin capacity*, $C(u, s)$, is the normalized transmission rate that client u can obtain in resource bin s, such that it meets the requirements on error probability, given the estimated channel quality. $C(u, s)$ is given in *bits per symbol*.

A matrix with $U \times S$ elements with $C(u, s)$ in the u, s position, is termed *capacity matrix*, and is denoted C.

■

Definition 4.2: *Average bin capacity $\bar{C}_u$*

The *average bin capacity* for client u, $\bar{C}_u$, is the bin capacity for client u, averaged over S resource bins:

$$\bar{C}_u = \frac{1}{S} \sum_{s=1}^{S} C(u, s). \tag{4.1}$$

■

Definition 4.3: *Average allocated bin capacity C_u*

The *average allocated bin capacity* for client u, C_u, is the bin capacity, averaged only over the resource bins that were allocated to client u:

$$C_u = \frac{\sum_{s=1}^{S} \delta(d_s - u)C(u,s)}{\sum_{s=1}^{S} \delta(d_s - u)}, \tag{4.2}$$

where $d_s \in [1, ..., U]$ is the scheduling decision for resource bin s, that is, an index number to the client selected to use resource s, and

$$\delta(x) = \begin{cases} 1, & \text{if } x = 0 \\ 0, & \text{otherwise.} \end{cases} \tag{4.3}$$

■

Each entry in the matrix C represents the number of payload bits per symbol

$$R = R_c log_2 M \tag{4.4}$$

for the appropriate modulation format, with M-ary constellation and rate R_c coding. The modulation is selected based on the predicted channel quality for each resource bin and client, to give the highest transmission rate compatible with the target bit error rate for the client.

Example 4.1: Bin capacities

For a resource bin s in channel u where uncoded BPSK modulation can be utilized, the bin capacity, $C(u,s) = 1$. If uncoded 8-PSK can be used, $C(u,s) = 3$. If 8-PSK with rate 1/3 channel coding can be used, then (4.4) gives $C(u,s) = \frac{1}{3}\log_2(2^3) = 1$.

We will use the capacity matrix C and the bin capacity $C(u,s)$ later, in Chapter 6 and Chapter 7, to represent the resource quality, in our proposed scheduling algorithms.

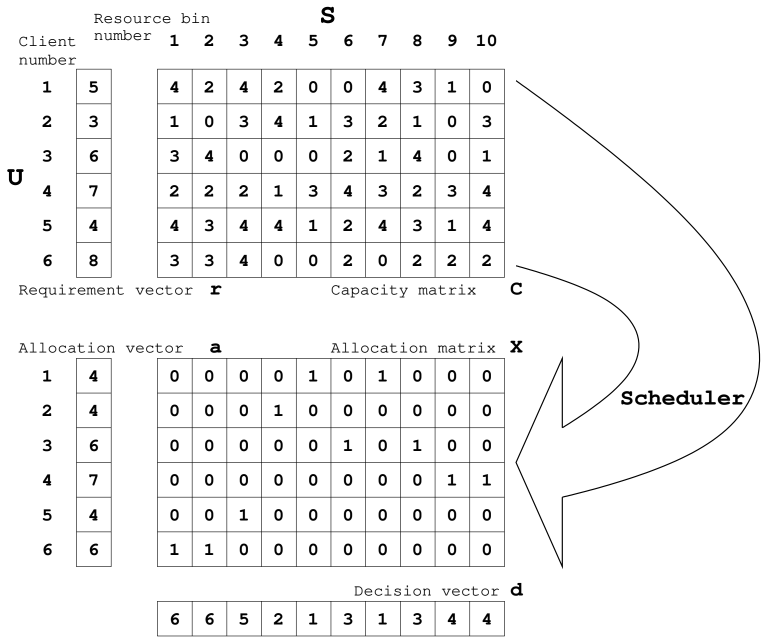


Figure 4.1: Example of a scheduling setup with six clients and ten resource bins. The service requirement vector at the top left contains the token level for each client. The capacity matrix at the top right contains the feasible modulation and coding formats for each user and bin, according to (4.4), that is, the bin capacity. At the bottom is the decision vector indexing the client number that has been scheduled for a particular resource bin. It can be translated into the binary allocation matrix X, and, through the capacity matrix (and Equation (4.6)), the resulting throughput for each client is found in the allocation vector, at the bottom left. In this example, the schedule was calculated with the CSD algorithm, presented in Section 6.3.4, and it can be seen that the allocation is not optimal. For example, we can see that if clients 1 and 4 swap their bins 5 and 9, then also client 1 obtains the required service, which is not the case in the current schedule.

4.2.2 Service Requirements

The required throughput is given by the current token level for each client. The tokens are normalized to represent the resource bin size used in the radio link layer. In Example 4.2 the token level is related to the capacity matrix (see Figure 4.1).

Example 4.2: Tokens, Resource Bins, and Data

If there is a service requirement of 12 (represented by 12 tokens in a client's bucket), it can either be satisfied by 12 resource bins with uncoded BPSK modulation ($R = 1$), or by 2 resource bins with uncoded 64-QAM modulation ($R = 6$).

Depending on the number of symbols within a resource bin, a token represents an absolute amount of data. In a system where a resource bin consists of e.g. 108 payload symbols, one token represents 108 data bits. 12 tokens then represents $12 \times 108 = 1296$ data bits.

In order to allow for short-term flexibility in the scheduling decision, admission control will maintain a certain level of tokens in the buckets, to ensure that it is the scheduler, and not the admission control, that decides how to distribute the resource bins.

The output from the scheduler is a vector with one entry for each resource bin ($1 \times S$), where each entry is a positive integer, the client number, that indicates the client selected to transmit in that particular bin. This vector is hereafter referred to as the *decision vector*, d. 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 resource bin. The "1" is in the location of the client that was allocated to that time-slot, and the "0"s are in the other locations. This matrix is denoted the *allocation matrix*, X. The decision vector is then given by

$$d = (1, 2, ..., U)\, X. \tag{4.5}$$

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

The resulting *allocation vector*, a, of size $1 \times U$ represents the resulting throughput for each client over the currently scheduled resource bin set. It is given by

$$a = diag\left(CX^T\right) \tag{4.6}$$

The resulting difference vector

$$w = r - a \tag{4.7}$$

represents the unfulfilled requirements and the unused resources (negative elements). In the example in Figure 4.1, all clients' requirements were not satisfied: Client 1 and 2 would need to make some exchange, in order to meet the requirements. Also, client number 6 was under-supplied. This particular schedule was found by the CSD algorithm which evidently could not find the optimal solution in this case. We will elaborate on this issue in Chapter 6, where the CSD algorithm is presented.

4.2.3 Complexity

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 [15], 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 finding a closed form solution [44]. We will instead resort to numerical techniques that, at least approximately, minimize a cost function J.

4.3 A Scheduled Communication System

As a target application, to test scheduling algorithms proposed in this thesis, we have selected a down-link dominated wireless communication system. Below, we outline the main functionalities required for realizing such a system. The flexibility in the system is based on a *feedback loop*, assuring corresponding actions at both ends of the wireless links:

1. The bin capacities are predicted for all available resources, may they be divided into frequency bins, time slots, antenna elements, polarization directions, spreading codes, or any other method for dividing the available resources, or combinations thereof. The prediction horizon is selected so the the delay of the scheduling feedback loop will not influence the scheduler decision.

2. All user predictions are communicated to a centralized resource scheduler. The more centralized the scheduler, the more holistic the resource optimization.

3. The result from the revenue-directed resource optimization, the scheduling decision, is communicated to the active clients, well in time for them to prepare for the execution of the scheduling decision.

4. All active clients take part in the radio reception in all resource bins, in order to collect channel quality estimates required in step 1 above, for the bin capacity predictions.

Thus, apart from the scheduling itself, a channel predictive scheduling system requires three main functionalities:

- Channel estimation, for coherent detection and also for step 4 above.
- Channel prediction, for step 1 above.
- Control signalling, for steps 2 and 3 above.

These issues, which will be discussed below, may result in difficulties that limit the practical value of using the channel prediction and scheduling approach in all cases. In [38, 26], general summaries of the involved trade-offs are given, together with case studies. A conclusion is that the network's spectral efficiency depends much on the imposed design constraints, and on the scenario under which it is tested.

4.3.1 Channel Estimation

Channel properties, such as received signal power and phase shift, can be estimated at the receiver. These estimates are normally utilized for channel equalization at the receiver[1] to mitigate the effects from the channel. Apart from channel equalization, the channel estimates are used to produce predictions of the channel properties in order to optimize the resource utililzation.

According to the results presented in [29], the channel estimates will contain errors that depend on

- the measurement noise,
- the current channel properties,
- the transmitted symbols used in the estimation (pilot or blind), and,
- the method used for the estimation.

For the block-based LS (least squares) estimation of flat fading channels, outlined in [29], the error can be partitioned into two main types of contributions:

- Errors due to measurement noise, and

[1]It is also possible to pre-equalize at the transmitter.

- bias errors, due to the block-wise constant approximation of a time-varying channel.

Measurement noise is reduced by adding more samples, whereas the bias components increase with an increasing length in time of the estimation interval. Lower received power will tend to decrease the SNR leading to worse estimations. This can be counteracted by using a longer estimation interval, but for fading time-varying channels, the bias contributions to the total error mentioned above will limit the useful length of the observation interval.

Downlink Channel Estimation

In order to obtain an estimate of a channel, a measurement has to be performed. Known pilot signals are transmitted and detected for this purpose. For the downlink channel it should not be a problem, since any mobile can participate in the reception of pilot symbols from the base station. So, for the downlink, a simple solution is to let the base station periodically broadcast a pilot signal. Then, all mobiles can estimate their downlink channel from this pilot. Furthermore, the estimates can be iteratively improved by means of decision-directed estimation, that uses also the detected payload symbols for the purpose of refining the channel estimate.

4.3.2 Channel Prediction

Based on historical channel states, and the current states, model based prediction can achieve acceptable predictions over fractions of a wavelength.

Example 4.3: Prediction Horizon

Let us assume that it is possible to predict the channel accurately for $\lambda/2$ meters, that is, half a wavelength. For a 2 GHz carrier,

$$\lambda = \frac{c}{f} = \frac{3 \cdot 10^8}{2 \cdot 10^9} = 0.15\text{m}.$$

Let us further assume that we would like to support mobiles travelling at up to 100 km/h, or, 27 m/s. Then, in order to have sufficiently accurate predictions, we may use a prediction horizon of

$$T = \frac{\lambda/2}{v} = \frac{0.15/2}{27} = 2.78\text{ms}.$$

Before arriving at the time instant of which we predicted the channel in Example 4.3, equal to 2.78 ms, a number of issues have to be handled. The most important ones are

- channel estimation,
- channel prediction,
- signalling of channel predictions to the scheduler,
- calculation of the schedule,
- signalling of the scheduling decision to the clients, and
- preparation at the clients for execution of the decision.

We believe that the most time-consuming task will be that of calculating the schedule. *Scheduling algorithms that provide good performance, yet have very low computational complexity, is key.* The design of such algorithms is a main contribution of this thesis and is the focus of Chapters 6 and 7.

4.3.3 Downlink Channel Quality Signalling

Downlink channel quality signalling refers to the transmission of downlink channel quality information, from the mobile terminals to the base station, required for downlink scheduling at the base station. The mobiles have to transmit to the base station an indication of the current or predicted channel properties. The information transmitted could be the predicted rate at which the mobile will be able to receive data from the base station. However, there may be many resource slots for which this information is required, depending on how the resources are partitioned.

There are indications [47] that for a scheduled system exploiting multiuser diversity, this information need not be very extensive to be valuable, and it may not necessarily require more than a few bits. Furthermore, not all active mobiles will need to transmit this information all the time, since relatively bad channels are less likely to be scheduled. About 10% *feedback load* has been suggested [42], meaning that a channel quality threshold should be applied so that on average 10% of the users are above the threshold, and are allowed to feedback their channel quality to the scheduler.

An approach more in line with the system we have chosen to study more closely (see Section 4.3.5), is to let *each client* communicate their $C(u, s)$, see Definition 4.1, to the scheduler, not for all s, but for a portion of them. The size of this portion depends on

- the bandwidth available for control signalling in the uplink,
- the revenue that a client is expected to be generating, and
- the number of simultaneously active clients, U.

One could set a target for the *total number* of communicated $C(u, s)$ to the scheduler be $H = h \cdot S$, where S is the number of available resource bins, and

$$1 \leq h \leq U \tag{4.8}$$

is the *feedback factor* chosen based on a trade-off between the available control signalling bandwidth, and the acceptable risk of having a resource bin that no client has claimed. An example of different choices of h is given in Example 4.4, next.

Example 4.4: Deciding the total amount H of feedback information

A down-link scheduled wireless communication system utilizes $S = 25$ simultaneously scheduled resource bins. These are to be shared among $U = 20$ currently active clients. Different values of h result in different number H of reported $C(u, s)$ to the scheduler:

- Choosing $h = 1$ results in $H = h \cdot S = 1 \cdot 25 = 25$ reported $C(u, s)$. This choice of h gives a high risk of having unclaimed resource bins, since there are 25 bins and totally only 25 claims, on average $25/20 = 1.25$ from each client.
- Choosing $h = 3$ results in $H = h \cdot S = 3 \cdot 25 = 75$ reported $C(u, s)$. This choice of h gives a lower risk of having unclaimed resource bins, since there are 25 bins and totally 75 claims, on average $75/20 = 3.75$ from each client, or, on average $h = 3$ claims for each resource bin. On the other hand, this choice requires more control signalling bandwidth.

Having chosen e.g. $h = 3$, or, totally $H = 75$ reports per scheduling round, the next step is to distribute them among the clients. One example of how to distribute them is given in Example 4.5.

Example 4.5: Deciding each client's amount of feedback information

We have decided to allow submission of a total of $H = 75$ resource bin reports to the scheduler. There are $S = 25$ resource bins to compete for, and $U = 20$ competing clients. In order to not constrain the scheduler more than necessary in making its decision, we choose to let the 15 most lucrative (in terms of cost efficiency) clients claim 4 resource bins each, and the remaining 5 clients 3 resource bins each, resulting in a total of

$$4 \cdot 15 + 3 \cdot 5 = 60 + 15 = 75 = H$$

reports.

A claim, or resource bin report, may then look like in Table 4.1.

Client ID,	$u =$	5	5	5	5
Resource bin ID,	$s =$	8	9	23	24
Bin capacity,	$C(u, s) =$	5	5	4	4

Table 4.1: A resource bin report, or claim, from client number 5. It has bin capacity 5 for resource bins 8 and 9, and bin capacity 4 for resource bins 23 and 24. The client number may be omitted from the report if this can be detected by other means, such as scrambling or spreading code.

4.3.4 Downlink and Uplink Schedule Signalling

Downlink schedule signalling refers to transmission of downlink scheduling information, from the scheduler at the base station to the mobile terminals, required for downlink data transmissions. Uplink schedule signalling refers to transmission of the uplink scheduling information, from the scheduler at the base station to the mobile terminals, required for uplink data transmissions. For the general case, a substantial overhead will be required to transmit the decision to the terminals. The base station can broadcast the scheduling decision, a decision that includes a session ID, indexed by the number of the resource bin, see Table 4.2 and Equation (4.9),

$$N = S \times \log_2 U, \tag{4.9}$$

where N is the number of bits in the scheduling decision for one scheduling cycle (time-slot), S is the number of scheduled resource bins, and U is the number of active sessions. This assumes that the *clients already know the rates* at which the data can be transmitted in the different resource bins, should they be selected for transmission by the scheduler. This information was previously created in the mobile terminals and then transmitted to the scheduler at the base station, in the form of a resource claim or capacity report, as outlined in Example 4.5.

Resource bin ID,	$s =$	1	2	3	4	...	S
Client ID,	$U \geq u =$	3	1	12	3	...	15

Table 4.2: Example of a scheduling decision transmission frame. In each resource bin, the information of which client that gets to transmit in that bin in the next cycle, is broadcasted. The client IDs are thus organized in an array indexed by the resource bin number, just like the decision vector d, defined in (4.5).

This requires that all active mobiles detect and decode this information all the time. The required amount of signalling needs to be properly addressed in the design process. For our target system, we will elaborate further on this topic in Section 4.3.5.

4.3.5 Conclusions and Outline of a Proposed System

Based on the reasoning carried out in this chapter, we shall now specify a system that uses fast resource scheduling. This is our target system, which will be discussed in detail in the case study, presented in Chapter 9. We shall only consider fast resource scheduling in the down-link.

In order to provide the required flexibility for the down-link, we choose to focus on an orthogonal frequency division multiplexing (OFDM) link. OFDM has the potential advantage of comprising a large number of densely spaced orthogonal subcarriers, that may be independently modulated and demodulated. The physical link that we choose to employ, is a number of 5 MHz carriers in the 1900 MHz band[2]. Each carrier is subdivided into 500 OFDM subcarriers. They are grouped into 25 frequency bins, each consisting of 20 subcarriers with a spacing of 10kHz. They are further subdivided in time, into time-slots of 6 symbols' duration. The symbol duration is $100\mu s$, which with a guard time of $11\mu s$ (used for a cyclic prefix), results in a symbol period of $111\mu s$. The carrier spacing of 10 kHz then results in time-frequency

[2]This choice is based on possible re-use of 3G spectrum allocations and equipment.

bins of 0.667 ms length and 200 kHz width. Out of the 120 symbols in each resource bin, 12 are used for pilots and control information, leaving a total of 108 symbols for data transmission in each bin, see Figure 4.2. These time-

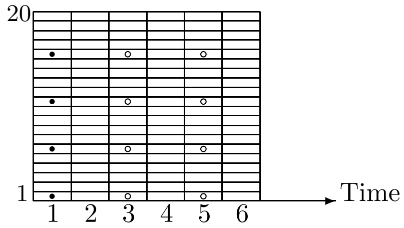


Figure 4.2: One of the time-frequency bins of the proposed system, containing 20 subcarriers with 6 symbols each. Known 4-QAM pilot symbols (black) and 4-QAM downlink control symbols (rings) are placed on four pilot subcarriers. The modulation format for the other (payload) symbols is adjusted adaptively. All payload symbols within a bin use the same modulation format.

frequency bins can be allocated separately and on-line to different clients, by the resource scheduler. The 25 time-frequency bins constitute one *scheduling frame*.

We repeat the calculations from Example 4.3, but with the new values corresponding to a prediction range of $\lambda/3$ with acceptable accuracy (NMSE of about 0.1, see [29, 30]) for a mobile speed of up to 100 km/h (or 27 m/s). For a carrier frequency $f_c = 1990$ MHz,

$$\lambda = \frac{c}{f_c} = \frac{3 \cdot 10^8 \text{ m/s}}{1990 \cdot 10^6 \text{ Hz}} = 0.151 \text{ m},$$

and a mobile speed of $v = 27$ m/s we can allow a prediction horizon of

$$T = \frac{\lambda/3}{v} = \frac{0.151/3 \text{ m}}{27 \text{ m/s}} = 1.86 \cdot 10^{-3} \text{ s} = 1.86 \text{ ms}.$$

This time horizon corresponds to almost three whole time-slots of duration 0.667 ms, in which we will have to

- generate the downlink channel predictions and calculate the corresponding bin capacities in resource bin 0
- then transmit the resource claims in the up-link control channel during resource bin 1.
- The scheduler will produce the schedule during bin 2, and

- broadcast the scheduling decision, embedded with the transmitted data, during bin 3, which is the predicted bin.

Thus, all active clients will have to decode all the transmitted control data, and then from the control symbols, identify which bins contain each client's designated data.

Although it may require a large amount of feedback bandwidth, we will in our case study (see Chapter 9) let all clients communicate all their bin capacities to the scheduler. Thereby we avoid the possible problem of unclaimed resources being wasted in our simulations.

However, it would be in place with an investigation of the required amount of feedback, or the feedback factor h, in (4.8), in order to trade a substantial reduction of the required feedback bandwidth for an acceptable probability of unclaimed resources, and a resulting loss of revenue. This is a topic for further research and will not be pursued further.

Chapter 5

Scheduling Algorithms

In this chapter we describe and define scheduling algorithms commonly referred to in the literature. Their main application area is within computer communications, but some algorithms also appear in computer processor sharing systems, or cashier queueing systems with multiple customers.

We have to bear in mind that different scheduling strategies may have different objectives. There are schedulers that aim at providing *fairness* among the served clients, whereas others aim at minimizing the *average waiting time*, or any other measure of performance relevant to the served clients.

We are primarily interested in scheduling algorithms that take into account

- varying service rates at the server, and,
- different service requirements among different clients.

Algorithms that fulfill these requirements are of potential use for data streams over wireless links with different QoS requirements. They are therefore here termed *Wireless QoS* scheduling algorithms. For completeness we will in this outline include also other types of scheduling algorithms that do not meet these requirements. The chapter is therefore divided into sections encompassing different types of scheduling algorithms. They are:

- Wireline Fair
- Wireless Fair / throughput maximizing
- Wireless QoS

It should also be said that, even though we divide the scheduling disciplines into distinct classes according to their applicability, there exist variations to the standard algorithms that extend their applicability into the other classes. One such example is given below, in Section 5.4.1.

However, we start with some necessary definitions.

5.1 Definitions

In order to explain the different scheduling algorithms we will need to define a number of terms commonly used in conjunction with scheduling.

Definition 5.1: *Backlogged client*

A *backlogged client* is a client with service requests in a queue, waiting for service.

■

5.2 Wireline Fair Scheduling Algorithms

We choose to present this type of algorithms for reference and for the purpose of introduction to the notion of *fairness*, and what shortcomings wireline algorithms meet, should they be applied directly onto wireless fading links. In the wireline scenario, the transmission link enjoys a constant capacity, $C(t) = C$, that the scheduler should assign to the different backlogged clients, whereas wireless links are subject to fading that is independent between clients.

5.2.1 Generalized Processor Sharing (GPS)

A resource scheduled according to Generalized Processor Sharing (GPS) [61, 14] will be shared among the clients in a way so that for any time interval, the resource will have been serving a backlogged client i at least as much as its guaranteed share, s_i, of the total service rate. The share for client c_i is expressed by

$$s_i = \frac{r_i}{\sum_{u=1}^{U(t)} r_u} C(t), \tag{5.1}$$

where r_i is the *rate weight* for client i, which dictates what portion of the service capacity $C(t)$, available at time t, should be given to client c_i. Furthermore, $C(t)$ is the total available capacity at time t, with $U(t)$ being the number of backlogged clients at time t. We illustrate how (5.1) can be used in the following two examples.

Example 5.1: Rate weight fairness example 1

If client c_1 has rate weight $r_1 = 1$ and client c_2 has rate weight $r_2 = 2$, then c_2 should receive twice as high a service rate than that of c_1. Moreover, if no other clients than c_1 and c_2 exist in the system at time t, then client c_1 should receive $\frac{1}{3}C(t)$, while client c_2 should receive $\frac{2}{3}C(t)$ of the available capacity $C(t)$ at time t.

Example 5.2: Rate weight fairness example 2

Let $W_i(t_1, t_2)$ denote the assigned bandwidth, or transmission rate, to client c_i during a time interval $[t_1, t_2]$. If $s_i(t)$ is allocated according to (5.1), and

$$W_i(t_1, t_2) = \sum_{t=t_1}^{t_2} s_i(t),$$

then

$$\left| \frac{W_i(t_1, t_2)}{r_i} - \frac{W_j(t_1, t_2)}{r_j} \right| = 0 \tag{5.2}$$

for any two backlogged clients c_i and c_j. Obviously (5.2) is fulfilled by the choices made in Example 5.1.

Equation (5.2) is the fairness criterion. It means that a client that needs service, will, during any short time interval $[t_1, t_2]$, at least get a share of the total service according to its pre-defined service rate share r_i. This can be guaranteed as long as the sum of all the guaranteed service rate shares does not exceed the total service rate.

GPS [61] is not really a scheduling algorithm by itself. We have chosen to present it since it is used in the scheduling community as a benchmark for how an idealized fair scheduler should behave. The GPS discipline requires a fluid model for the traffic that is served, meaning that the traffic must be infinitely divisible, and that all backlogged clients can be served simultaneously. Since this is not the case for real data traffic, the GPS cannot be realized exactly, but its behaviour can be approximated in different ways.

5.2.2 Weighted Fair Queueing (WFQ)

Weighted Fair Queueing (WFQ)[14, 12] tries to assign resources to the clients according to their pre-defined service rate requirements. It is considered a packet-by-packet approximation of the GPS discipline, and has thus also been called PGPS, for *Packet-by-packet Generalized Processor Sharing.* The idea is to calculate the finish time, $F(p_i^k)$, for transmitting packet number k to client i, had it been served by a GPS based server. Then, the incoming packets are served in increasing order of this finish time. The finish time is calculated as

$$F(p_i^k) = S(p_i^k) + \frac{L_i^k}{r_i}, \tag{5.3}$$

where L_i^k is the length of the kth packet for client i, and the start time $S(p_i^k)$ is calculated as

$$S(p_i^k) = \max[V(A(p_i^k)), F(p_i^{k-1})], \tag{5.4}$$

where $A(p_i^k)$ is the arrival time of packet p_i^k to the queue, and $V(t)$ is the simulated GPS system *virtual time* at time t, as defined below.

Definition 5.2: *Virtual time*
The *virtual time* $V(t)$ of a simulated GPS queueing system is given by

$$\frac{dV(t)}{dt} = \frac{1}{\sum_u^{U(t)} r_u} C(t),$$

where $U(t)$ is the number of backlogged clients, and r_u is each client's rate weight.

■

The finish time calculation in (5.3) simply adds the expected processing time L_i^k/r_i to the start time $S(p_i^k)$.

A WFQ system requires that the virtual time $V(t)$ is updated for each backlogged client, in order to decide which client to serve next. Furthermore, it is required that a finish time $F(p_i^k)$ is calculated and stored for each job (data packet) in the queueing system. The virtual time $V(t)$ is thus a crucial part of the criterion for the scheduler.

5.2.3 Worst-case Fair Weighted Fair Scheduling (WF^{2}Q)

Different improvements to the WFQ discipline have been suggested, both in terms of performance and complexity. One of them is the Worst-case Fair Weighted Fair Queueing discipline (WF^{2}Q) [12], that we describe next by means of comparison with the WFQ discipline. In the WFQ discipline, the client to be served is selected among the backlogged clients according to the smallest calculated finish time $F(p_i^k)$. It is argued in [12] that a better approximation to the GPS discipline is accomplished if, instead of considering all backlogged clients, only the clients with packets (jobs) that would have been starting to receive service under the GPS discipline, are considered. That is, only packets are considered that fulfill

$$\{p_i^k | S_{i,GPS}^k(p_i^k) \leq V(t) \leq S_i^k(p_i^k)\}, \tag{5.5}$$

where $S_{i,GPS}^k(p_i^k)$ is the time at which the packet p_i^k would have started to receive service under the GPS discipline, $V(t)$ is the GPS virtual time from Definition 5.2, and $S_i^k(p_i^k)$ is the starting time as defined in equation (5.4) above. Among these jobs, the one with the earliest finish time is selected for service.

5.2.4 Round Robin (RR)

The Round Robin algorithm (RR) is very fair and simple when the service rate is constant [69]. The backlogged clients take turns in utilizing the service for a short period of time, or for a pre-defined amount of work, in a cyclic fashion. RR may also serve different clients differently, according to a rate weight, as for the WFQ discipline.

Unfortunately, this approach does not fit well into the scenario where the server (the wireless link) has a varying work rate (fading channels), and more importantly, when the server offers different work rates for different clients at the same time (independendt channels).

5.2.5 Summary

Wireline fair algorithms will not exploit the wireless channel variations in an advantageous way. In a wireless scenario, the drawback of the wireline fair algorithms presented in this section, is that they do not take into account the varying and different service capacities that the different clients experience due to fading. Therefore, we expect that other algorithms, taking these variations into account, would perform much better. We will discuss one such group of algorithms next.

5.3 Wireless Fairness Throughput Scheduling Algorithms

This type of scheduling algorithms exploit the wireless channel variations that impose varying and different service rates for different clients. They also aim at giving a fair share of the available capacity to the different clients. The actual meaning of *fairness* in the use of wireless transmission resources is ambiguous, and it still receives considerable research attention. See [56] and references therein.

The wireless channel variations offer the possibility to achieve *multiuser scheduling gain*, should the variations be efficiently exploited. The multiuser scheduling gain results from the possibility to utilize the wireless channels at higher transmission rates than their average possible rates, by utilizing temporarily good transmission conditions, and avoiding temporarily bad channel conditions, while still utilizing all channel resources all the time. This is what *opportunistic scheduling* algorithms do, by efficiently sharing the resources among multiple users. Wireless fair scheduling algorithms are one such type of algorithms.

5.3.1 Proportional Fair Scheduling (PF)

The Proportional Fair Scheduling algorithm (PF) [45, 51], allocates the resource to the user that can transmit at the highest rate, relative to its average achieved throughput during some window of past transmissions.

In its most basic form, the PF algorithm assigns the resource to the backlogged client indicated by

$$\arg\max_i \frac{(C/I)_i(t)}{T_i(t)}, \tag{5.6}$$

where $(C/I)_i(t)$ is the current carrier-to-interference ratio for client i at time t, and $T_i(t)$ is the achieved data throughput for client i, in a limited time window, up to time t. PF scheduling is resource efficient, in that it utilizes the link (the server) when its service rate is estimated to be high. It is also fair in the sense that clients that have received worse service for some time, will be compensated as their weights in (5.6) increase as the denominators $T_i(t)$ decrease.

Remark 5.1: *PF variants*

There are variants of (5.6). In e.g. [67, 75], instead of normalizing with the average *attained* throughput $T_i(t)$, normalization is performed with the average *attainable* throughput, which results in the scheduler assigning the resource to the client with the best resource quality as compared to its average resource quality.

■

Remark 5.2: *PF favourizes large channel variations*

In [45] it is observed that clients with channels subject to larger variation, all other things being equal, enjoy a better service (higher throughput using less resources), than clients with less varying channels, under the PF algorithm.

■

Remark 5.3: *PF in CDMA2000*

According to [51], the PF algorithm is used in the 1xEV-DO system, which is the second phase of evolution of the CDMA2000 standard. [1] In 1xEV-DO, CDMA has been abandoned as the method for multiplexing multiple users, but is kept for the purpose of spreading. Instead, TDMA is used for multiplexing.

■

5.3.2 Score Based Scheduling (SB)

A more general approach, that efficiently combats the shortcomings mentioned in Remark 5.2 of the standard PF algorithm, does not only look at the average of the recent transmission rates, but at the whole distribution. The distribution is used to normalize each client's current possible throughput [62, 18]. The solution in [18], called Score Based (SB) scheduling, is as simple as it is fair:

- Let $r_i(t)$ be the possible transmission rate for client i at time t.
- Store each client's W most recent possible transmission rates $\{r_i(\tau)|t-W+1 \leq \tau \leq t\}$ in a list sorted in ascending order of r_i, where W is the size of a time window.
- Now let $s_i(t)$ be the position index in each list to each client's current transmission rate $r_i(t)$.

[1] The first phase of evolution of CDMA2000 is called High Data Rate (HDR).

- Assign the channel at time t to the client i with the maximum $s_i(t)$.

This algorithm enjoys a very simple implementation. It works directly on the achievable service rates, and does not require any weight calculations or prioritizations. Moreover, it takes into account the statistical distribution, without making any assumptions about its shape.

5.3.3 CDF-based Scheduling (CS)

The Cumulative Distribution Function (CDF) based Scheduling (CS) approach in [62] is that of assuming a statistical distribution for the possible service rates for each client, then assigning the resource to the client that, according to the distribution and the current possible rate, has the least probability of obtaining a higher rate. This approach also combats the shortcomings mentioned in Remark 5.2.

For the Gaussian case, the algorithm assigns the resource to the client that maximizes:

$$\frac{R_i(t) - m_i}{\sigma_i}, \tag{5.7}$$

where $R_i(t)$ is the possible rate at time t for client i, m_i is the mean of the Gaussian distribution for client i, and σ_i^2 is its variance.

Remark 5.4: *Distribution shape assumption*

The CDF rule requires, or assumes, knowledge of the distribution function behind the achievable rates, whereas the SB rule in Section 5.3.2 does not make any assumptions about this.

■

Potentially, one could build in prior information into the CDF of the CS algorithm to make it more powerful. This could be e.g. that the mobile is approaching the base station, and will therefore expectedly experience an increased channel quality. This would be to look ahead, rather than back to stored data, as is done in the SB algorithm.

5.4 Wireless QoS Scheduling Algorithms

The algorithms proposed in this thesis all fit into this category. The aim is to take the channel properties into account, in order to efficiently utilize the resources when they offer good service, and at the same time pay attention to throughput and delay requirements from the clients, in order to offer attractive and predictable services.

5.4.1 Modified Proportional Fair Scheduling (MPF)

An extension to the standard PF algorithm is given in [10], namely the Modified Proportional Fair (MPF) algorithm. It is modified in order to support delay sensitive clients. The modification is as follows:

- Define different types of clients with different requirements on delay
- For each delay constrained client i, define a threshold delay value, τ_i
- For each delay constrained client i, maintain an actual delay value, d_i
- For each client, define a prioritization factor K_i that is inversely proportional to its current channel quality:

$$K_i = \frac{(C/I)_{max}}{(C/I)_i} p_i, \tag{5.8}$$

  where

  - $(C/I)_{max}$ is the maximum of all clients' current channel conditions, and
  - $p_i \geq 1$ is a prioritization factor that assigns a priority to client i reflecting the importance of it meeting its delay requirements.

- Assign the resource to the client i that maximizes

$$P_i = \begin{cases} \frac{(C/I)_i}{T_i} K_i & , d_i > \tau_i \\ \frac{(C/I)_i}{T_i} & , d_i \leq \tau_i \end{cases} \tag{5.9}$$

Thus, the delay support is in this case provided in two steps, activated by the current delay experienced by the client d_i passing its allowed threshold value τ_i:

1. Eliminate the proportional part from the PF algorithm by cancelling the channel dependent factor $(C/I)_i$ in (5.9)
2. and multiply by a factor p_i that reflects its delay support importance.

Intuitively, MPF gives an increased absolute priority to clients that have exceeded their delay constraints. When multiple clients enjoy this prioritization, they will internally be scheduled according to their prioritization factors.

Remark 5.5: *Advantages and shortcomings of MPF*

At the same time as the MPF algorithm increases the priority of a delay sensitive client in order to meet its delay requirements, it also gives up the possibility to achieve a scheduling gain due to the channel variations.

■

However, this is a sound behaviour if delay is an issue that is prioritized and payed for.

5.4.2 Modified Largest Weighted Delay First (M-LWDF)

The Modified Largest Weighted Delay First (M-LWDF) [67, 4, 5, 6] stems from the Largest Weighted Delay First (LWDF) [76] algorithm, originally intended for QoS provisioning over fixed-rate channels. The basic LWDF algorithm is thus not adequate for time-varying channels, such as wireless fading channels. The idea in LWDF is to assign the channel to the client that maximizes the ratio between the current delay or waiting time, $\hat{w}_i(t)$, experienced by the longest waiting packet in each queue, and a QoS-related factor $\alpha_i > 0$, assigning lower α_i to clients with lower delay tolerance. LWDF assigns the resource to the client i according to

$$\arg\max_i \hat{w}_i(t)/\alpha_i. \tag{5.10}$$

The modification to the LWDF rule introduces the varying service rate for different clients. Apart from providing QoS support, the M-LWDF algorithm then takes the channel capacity variations into account: The M-LWDF algorithm selects for transmission the packet that maximizes the value

$$\frac{p_i \hat{w}_i(t)}{c_i(t)}, \tag{5.11}$$

where $\hat{w}_i(t)$ is the delay of the first packet in queue i, $c_i(t)$ is a weighting function proportional to the required transmission power, thus depending on the current channel state. The QoS support comes through the factor

$$p_i = a_i \bar{c_i}, \tag{5.12}$$

where $\bar{c_i}$ is the time average of $c_i(t)$ for client i, and a_i is a QoS parameter, suggested to have the value

$$a_i = -\frac{\log(\delta_i)}{T_i}, \tag{5.13}$$

where δ_i is the desired probability to fulfill the delay requirement T_i.

The algorithm is basically the same as the *Max Weight* (MW) algorithm described in [78], if the QoS-dependent weight factor a_i is omitted.

Note that the current channel capacity is introduced indirectly in the criterion (5.11), as the required transmission power, $c_i(t)$, in order to uphold a certain SNR. This approach is taken since the scheduling algorithm is proposed for a CDMA system, that relies on power control.

5.4.3 Exponential Rule (ER)

The Exponential Rule (ER) [67] is based on the M-LWDF rule (Section 5.4.2), but has more similarities with the MPF rule (Section 5.4.1). Following the ER rule, the channel is assigned to client i according to

$$\arg\max_i \frac{p_i}{c_i(t)} \exp\left(\frac{a_i \hat{w}_i(t) - \overline{a\hat{w}}}{1 + \sqrt{\overline{a\hat{w}}}}\right), \tag{5.14}$$

where the same notation is used as for the M-LWDF rule above.

Remark 5.6: *Similarities with PF*

The factor $\frac{p_i}{c_i(t)}$ in (5.14) is basically a weighted PF rule. This is understood if we consider the normalization of the PF rule, mentioned in Remark 5.1. Instead of normalizing with respect to the average *achieved* throughput $T_k(t)$ in (5.6), we normalize with the average *possible* throughput, which is proportional to $1/p_i$ in (5.14).

■

The thinking behind the ER rule is to let the (weighted) PF discipline control the scheduling as long as no delays grow too large. If any delay departs from the average, then the exponential factor will be "activated", and increase the priority for the client with the larger delay. The difference to the MPF rule (recall that MPF introduces a prioritization factor into PF when the delay grows too large) is that ER has a more graceful adaptation from PF, to a less channel dependent, more delay intolerant scheduling rule.

5.5 Summary

In Table 5.1 a summary of the properties of the presented scheduling algorithms is given. The different fields in the table answer the question of whether the algorithms, in some relevant way, take into account

1. *Channel* state information: The varying and different service rates that different clients experience, due to the fading channel properties.

2. *Queue* state information: The amount of queueing jobs (data packets, bytes) in the different clients' queues. This is required for bounded delay provisioning.

3. *External priority*: Externally assigned weights that reflect the relative importance of serving a client.

4. *Fairness* among users: Avoids service starvation for clients due to misbehaviour from other clients.

Algorithm	Channel state	Queue state	External priority	Fairness
GPS	No	No	Yes	Yes
WFQ	No	No	Yes	Yes
WF^2Q	No	No	Yes	Yes
RR	No	No	No	Yes
PF	Yes	No	No	Yes
SB	Yes	No	No	Yes
CS	Yes	No	No	Yes
MPF	Yes	Yes	Yes	Yes
M-LWDF	Yes	Yes	Yes	No
MW	Yes	Yes	No	No
ER	Yes	Yes	Yes	Yes

Table 5.1: Summary of the presented scheduling algorithms. The cells say whether or not the algorithm takes into account the channel state, the queueing state, any external priority, or achieves fairness in some sense.

Among the presented algorithms, MPF, M-LWDF and ER all provide channel and queue state dependence, and they can also incorporate externally assigned priorities for different QoS requirements. They are therefore the most interesting algorithms to compare our proposed scheduling algorithms to. Furthermore, because of its improved fairness and simple implementation, the SB rule serves as inspiration for our proposed score based scheduling algorithm, presented in Section 7.4.2.

However, the presented scheduling algorithms that take into account the channel quality, only look at the *instantaneous* channel qualities, whereas the methods that we will propose utilize *predictions* of future channel qualities. Furthermore, the algorithms presented in this chapter only schedule the *next transmission* by evaluating some priority function, whereas our proposed scheduling methods will utilize the predicted channel qulities in order to schedule *multiple future transmissions*, thereby enabling a more holistic optimization.

Chapter 6

Power-n Scheduling Criteria

We have seen that admission control and its entities operate on time-scales at or below that of the slow fading and shadow fading. According to the long-term variations of the channels, it shall adapt the load that the scheduler will have to handle. We will now concretize ideas from Chapter 4, and turn to the small-scale fading properties of the wireless channels. The aim is to find ways for the communicating nodes to exploit them. We will assume that we have accurate channel quality predictions that in advance allow us to evaluate the benefit of assigning a specific channel resource to serve a client.

The choice of criterion is an issue for the network operator and its policies. It encompasses decisions on fairness and service requirements for different traffic classes. We will start from a rather general criterion for schedule optimization in (6.1), then elaborate on a special case of it, namely the quadratic critetrion (6.2). In order to create an on-line scheduling algorithm, we shall derive approximations to the minimization of the quadratic criterion, namely two variants of a linear approximation, that allow us to implement fast algorithms for finding near-optimum soultions. An important step in the algorithmic simplification will be a change of variables that allows the optimization to be performed individually for each resource bin.

Assuming that we have an admission control algorithm that shapes the offered traffic so that we have an output buffer with a controlled inflow rate, a reasonable criterion to minimize by the scheduler would be:

$$J = \sum_{u=1}^{U} P_u |r_u - a_u|^n \tag{6.1}$$

In (6.1), a_u is the amount of scheduled data blocks per client u, and r_u is the required amount of data blocks per client (the amount of queueing tokens[1] in the output buffer). The values a_u are constrained by the capacity matrix C, c.f. Definition 4.1 and presented in Figure 4.1, via the scheduling decision vector d.

Before we proceed, recall from Chapter 2 that the pricing models used for each client will cause the admssion policy (AP) to set token rate limits, and for delay sensitive clients, also token bucket level targets, for the service level controller (SLC) to uphold for each client. The SLC will then strive to attain the bucket target levels by increasing or decreasing the token rates, within the limits set by the AP, into the buckets. The token bucket level attained after each SLC control action is r_u. It here represents the required amount of data to be served for each client.

We thus have a cascade controller with three loops, where the inner loop is governed by the scheduler, based on the bucket levels, r_u, provided by the SLC. This chapter, and the next, will focus on this inner loop.

Proceeding with the criterion (6.1) we note that it is a function of the amount of data tokens left in the output buffers after the scheduling decision has been executed. By adjusting the exponent $n > 0$ and the weighting factors $P_u \geq 0$ we can control how the different buffer levels are affected. The use of the weighting factors P_u, and their relation to priorities (see Definitions 6.1 and 6.2 on page 90), is discussed further in Section 6.1.1. The term $|r_u - a_u|^n$ acts as an *internal priority* function (see Definition 6.2 on page 90).

Two interesting special cases for the exponent n are $n = 1$, and $n \to \infty$:

- For $n = 1$, the minimization of (6.1) yields a maximum throughput policy, meaning that the level in the output buffer will not influence the scheduling decision. Only the possible attainable throughput, due to the channel quality, would be taken into account.

- In the asymptotic case $n \to \infty$, the output buffer levels are dominating, making the scheduler always choose for transmission the queues that would minimize the largest of the buffer levels.

For othere values of n we can make the following qualitative observations:

- Values $n > 1$ result in a compromise between the two extremes, taking both buffer levels and channel quality into account.

[1]One token corresponds to the right of transmitting one data block, that in this case is the amount of data that a time-frequency bin can transmit when using BPSK modulation. This relation is explained in Example 4.2 on page 61.

- For values $0 < n < 1$, we achieve a policy that would prioritize streams with small buffer levels higher than large buffers, yielding something similar to Shortest Job First scheduling [69].

From this discussion we conclude that (6.1) is a flexible criterion capable of addressing both throughput and buffer levels, and at the same time account for SLC-induced external priorities. We illustrate these qualitative properties in Example 6.1 next.

Example 6.1: Different n with equal capacities $C = [c_1 \;\; c_2]^T = [1 \;\; 1]^T$

Let us consider a simple case where we have two clients with one output buffer each, numbered 1 and 2. Furthermore, we have only one resource bin that we have to assign exclusively to one of the two clients. The clients' buffer levels are 41 and 40 for client 1 and 2 respectively. According to the channel predicition, both clients may utilize the single channel resource at a rate of 1 (yielding a capacity matrix C with two rows and one column $C = [1 \;\; 1]^T$), meaning that either client may remove one queueing item from his buffer. It is now up to the scheduler to decide which one, using Equation (6.1), with $P_u = 1$ for simplicity. Trying some different values for n we obtain the following decisions:

- $n = 1$ results in (6.1) giving the values

  $$J_1 = |41 - 1|^1 + |40 - 0|^1 = 40 + 40 = 80$$

  and

  $$J_2 = |41 - 0|^1 + |40 - 1|^1 = 41 + 39 = 80.$$

  Since the cost function values are equal, the decision can not be done by the internal priority, and some arbitration is necessary, by e.g. random choice.

- $n = 100$ results in (6.1) giving the values

  $$J_1 = 40^{100} + 40^{100} = 0.32 \cdot 10^{161}$$

  and

  $$J_2 = 41^{100} + 39^{100} = 1.9 \cdot 10^{161}.$$

  Since $J_1 < J_2$, the scheduler will choose J_1 which is minimal, and thus allocate the resource to client 1. Then the decision vector will have one element (corresponding to this particular resource bin) containing the index to the selected client, $d = [1]$. The allocation vector will have two elements (one for each client) $a = [a_1 \;\; a_2] = [1 \;\; 0]$.

- $n = 0.5$ results in (6.1) giving the values

$$J_1 = 40^{0.5} + 40^{0.5} = 12.649$$

and

$$J_2 = 41^{0.5} + 39^{0.5} = 12.648.$$

Since $J_2 < J_1$, the scheduler will choose J_2 which is minimal, and thus allocate the resource bin to client 2.

We observe from Example 6.1 that if we have equal capacities and priorities for two clients, a large exponent ($n > 1$) will make the scheduler select for transmission the client with the larger buffer, whereas a small exponent ($0 < n < 1$) will make the scheduler choose the client with the smaller buffer. The choice of n will thus have a significant impact on the decisions made by the scheduler.

In Appendix B.1 we present a discussion on requirements on a scheduling criterion, for maintaining *stable* queues. As we show in Appendix B.1.1, stability requires that $n > 1$ in (6.1).

For the rest of the chapter we shall focus on the use of $n = 2$, thereby having a quadratic scheduling criterion. The reason for elaborating on the quadratic criterion is that we want a qualitative scheduling behaviour between the two extreme behaviours represented by $n = 1$ and $n \to \infty$. We want the scheduler to take into account both the buffer level and the channel quality, in order to make resource efficient scheduling decisions. Moreover, choosing the exponent $n = 2$ gives us the opportunity to proceed with a very simple implementation, in the linear approximation outlined below. Thus, both the desired qualities in the discrimination obtained by the criterion, as well as algorithmic complexity issues, lie behind this choice.

6.1 The Quadratic Scheduling Criterion

We hereby assume that we wish to minimize the sum of squares ($n = 2$) of the remaining contents in each buffer, each squared buffer term weighted by an arbitrary factor P_u, specific for that client u. The meaning and usage of the weighting factor P_u will be described further in Section 6.1.1.

With $n = 2$, (6.1) becomes[2]

$$J = \|r - a\|_P^2 = \sum_u P_u (r_u - a_u)^2 = \sum_u P_u \left(r_u - \sum_s C(u,s)X(u,s) \right)^2, \tag{6.2}$$

where $X(u,s)$ is the (u,s) entry in the binary allocation matrix X for client u at resource bin s, r_u is the uth entry in the requirement vector r, that is, client u's required bandwidth according to the token bucket level, see Figure 3.1 and Figure 4.1. If $X(u,s) = 1$, then a resource of capacity $C(u,s)$ is allocated to client u in bin s. Recall that $C(u,s)$ represents the highest possible allowed modulation level for client u in resource bin s (see Definition 4.1 on page 58). In (6.2), the summation

$$\sum_s C(u,s)X(u,s) = \left(CX^T\right)_{uu}$$

corresponds to calculating the allocation a_u in (4.6) for each client u. This is done for each client by summing the resources $C(u,s)X(u,s)$ allocated to that client, over all the resource bins s involved in the scheduling.

The cost function (6.2) will penalize large deviations from empty buffers. More specifically, reducing a large buffer content with a number of bytes will be preferred to reducing an almost empty buffer with the same amount of bytes.

The value of the criterion J depends on a_u, via X, through the scheduling decision, d. The vector d is of length S, c.f. (4.5), the number of available resource bins, and for each resource bin $s = [1, ..., S]$, it contains an index representing the client u chosen to use resource bin s,

$$d_s = u, \qquad u \in \{1, ..., U\}, s = 1, ..., S.$$

For example, a decision vector $d = [2, 5, 3, 12]$ means that the four resource bins 1, 2, 3, and 4 are allocated to client 2, 5, 3, and 12, respectively. Thus,

$$a_u(d) = \sum_{s=1}^{S} \delta(u - d_s) \cdot C(u,s), \qquad u = 1, ..., U, \tag{6.3}$$

where $\delta(x)$ is a unit impulse function, having the value 1 for $x = 0$, and being zero otherwise.

[2]The special case $n = 2$ is also discussed in Equation (6.2) in [32].

We may then express the criterion J as a function of d:

$$J(d) = \sum_{u=1}^{U} P_u(r_u^2 - 2r_u a_u(d) + a_u(d)^2). \quad (6.4)$$

This is also an explicit way of describing what the scheduling is all about: Finding the vector d that minimizes J. Thus, the scheduling decision can be expressed as

$$\begin{aligned}\hat{d} &= \arg\min_d J(d) \\ &= \arg\min_d \sum_{u=1}^{U} P_u(r_u^2 - 2r_u a_u(d) + a_u(d)^2). \quad (6.5)\end{aligned}$$

6.1.1 The Weighting Factor

The weighting factor P_u can be subdivided into two factors; P_{ui} and P_{ue}. P_{ui} will be used to *balance the internal priority* (see Definition 6.2 below) by compensating for the average channel conditions, while the *external priority* P_{ue} (see Definition 6.1 below) will reflect the delay requirements of the different clients.

Definition 6.1: *External Priority*

The *External Priority*, P_{ue}, defines a high-level relative importance of serving client u. It is intended to reflect the expected revenue obtained by serving the client in the long run.

■

Definition 6.2: *Internal Priority*

The *Internal Priority*, P_{ui}, defines a low-level relative importance of serving client u. It is intended to reflect a short term calculated preference of serving a specific client with a specific resource slot.

■

External Priority, P_{ue}

As a rule of thumb for handling delay requirements, we will use a larger P_{ue} when we strive to maintain a smaller delay for client u, thereby weighting up that term in (6.2). When comparing alternative schedules, the scheduler will then find that choosing a schedule in which it allocates more resources to a client u with a high external priority, P_{ue}, will result in a smaller criterion (6.2) than for the other possible schedules.

Example 6.2: External priority usage

The throughput admitted for client u by the admission control, I_u (e.g. in megabytes per second, MB/s), times the delay of the packet that has spent the longest time in the buffer, t_u (e.g. in milliseconds), gives a value,

$$\hat{r}_u = I_u \cdot t_u, \tag{6.6}$$

e.g. in kilobytes, that approximates the output buffer level for the data belonging to client u (this relation is known as Little's Law [44]).

Using Equation (6.6), we, or rather the AP, may also calculate a token bucket level $\hat{r}_u$ that, given the admitted rate I_u and a target delay t_u, serves as target level for the SLC, in order to accomplish the delay t_u.

In order to normalize this value in the criterion, and give all clients that require it, delays proportional to their target delays, we will select the inverse of this target token bucket level as external priority,

$$P_{ue} = \frac{1}{\hat{r}_u}. \tag{6.7}$$

The unit for P_{ue} will thus be $1/bit$ when dealing with bit-streams, while it may be dimensionless in a token-based scheduler.

Remark 6.1: *Target $\hat{r}$ appears twice*

Note that the target token bucket level $\hat{r}$ will be used in two places, namely as a reference token bucket level for the SLC, e.g. as $q(t)$ in (3.3), and as a weighting factor in (6.7).

■

Internal Priority Balancing, P_{ui}

If a client is far from the base station, then it is likely that the predicted average bin capacity, $\bar{C}_u$, is much lower for that client, than for a client near the base station. This would make the far client have small values a_u in the criterion. For P_{ui} we will then choose a larger value, the worse the average channel capacity for that stream is. As a rule of thumb, $P_{ui} = 1/C_u$, where C_u is the average *allocated* bin capacity[3], that is, the transmission rate that a client u experiences during the time when it accesses the channel. Thus the unit for P_{ui} will be $1/bit$, as well when dealing with bit-streams (or dimensionless when dealing with tokens).

Remark 6.2: *P_{ui} does not prioritize*

If no compensation for the average capacity is made by the service level controller (SLC) by setting the weight P_{ui}, a client with a bad channel would "automatically" be compensated by its growing buffer level, r_u, giving it an increasingly large criterion value, at the expense of a longer delay. In order to keep the buffer levels more equal, regardless of the average channel quality, the SLC can compensate the *client* for this effect by assigning a higher P_{ui} for a client with a bad average channel. Thus, compensation with P_{ui} should not result in more resources being spent on client u. Instead, we will let the SLC (see Section 3.3) handle the potential problem of increased cost for serving client u, by adjusting the admitted rate I_u, to reflect the changed average channel conditions.

■

Example 6.3: Internal priority balancing

Assume that we have a scheduler with two active sessions. One session corresponds to a mobile client A, that resides very near the base station. It will have a high average SNR value, and thus a high average capacity, C_{As}, for all his resource bins s. The other session belongs to client B, far from the base station, consequently having a low average capacity, C_{Bs}, for all his resource bins s. To compensate client B for his bad average channel conditions, we will use $P_{Bi} > P_{Ai}$, weighting up the term for client B in (6.2). On average, we aim at

$$P_{Ai} \cdot C_A = P_{Bi} \cdot C_B$$

in order to maintain equal buffer levels. The consequence of *not* compensating client B, would be that the queue (r_B in Equation (6.2)) corresponding to client B's session would grow until the queue size (or the output buffer level) would compensate for the bad channel, thereby increasing the delay

[3]The average allocated bin capacity C_u is defined in Definition 4.3 on page 59.

for client B. Since we wish to design a system that offers certain services according to SLA:s, we do not want the channel quality fluctuations to directly influence the service quality. Therefore, the SLC compensates client B with P_{Bi} in order to balance the services.

An alternative or complementary way to also compensate the *system*, instead of just compensating the client, is to admit a lower data (or token) rate I_u to a client with a bad average channel. Doing this alongside with the client compensation described above, we would assure that the compensated client does not use much more resources than other clients with better average channels. By letting the SLC choose a combination of rate reduction and average channel compensation, the operator can trade system capacity for client delay and throughput.

Example 6.4: Combining external priority P_{ue} with internal priority balancing P_{ui}

Having a balanced internal priority, using P_{ui}, the scheduler will work towards minimizing the buffer levels for all clients, while keeping the buffer levels equal. Introducing the external priority P_{ue} from (6.7), will then result in a criterion that will promote schedules that

1. allocate resources to efficiently serve the clients (because of the internal priority),
2. disregard long-term inequalities between different channels (because of P_{ui}), and
3. maintain proportional delays according to the requirements (because of P_{ue}).

A properly weighted scheduling criterion, J, reflects how well the scheduler is currently performing. By properly weighting the contributions from the different streams to the criterion, we can monitor whether the delays required by the SLA:s are maintained or broken. We can design control laws, based on the criterion (6.2), for the admission control algorithm to take action in order to make it possible for the scheduler to fulfill the delay requirements in the SLA:s. An increasing value of J indicates that the current load is increasing (or that the transmission capacity is decreasing), and the admission algorithm must adjust the offered work load mix. In other words, the criterion value is an indirect measure of the queueing delay, and the service level controller can regulate this by adjusting the input flows.

6.2 Linearization of the Quadratic Criterion

The terms r_u^2 in (6.5) will not affect which d minimizes the sum, since they are independent of d. Therefore we can remove these terms from the sum in (6.5) without loss of generality:

$$\begin{aligned} \hat{d} &= \arg\min_d J(d) \\ &= \arg\min_d \sum_{u=1}^{U} P_u(-2r_u a_u(d) + a_u(d)^2). \end{aligned} \tag{6.8}$$

Now, examining equation (6.8), keeping in mind that $J(d)$ is monotonically decreasing with an increasing a_u when $a_u \leq r_u$ and that $a_u \geq 0$, we can perform the following approximation: Replace one a_u in the last term of each paranthesis by r_u on the interval $0 \leq a_u \leq r_u$,

$$\begin{aligned} \hat{d} &= \arg\min_d J(d) \\ &= \arg\min_d \sum_{u=1}^{U} P_u(-2r_u a_u(d) + a_u(d)^2) \\ &\approx \arg\min_d \sum_{u=1}^{U} P_u(-2r_u a_u(d) + r_u a_u(d)) \\ &= \arg\min_d \sum_{u=1}^{U} -P_u r_u a_u(d) && (6.9) \\ &= \arg\max_d \sum_{u=1}^{U} P_u r_u a_u(d). && (6.10) \end{aligned}$$

This linearization will *mostly* maintain the discrimination in d since the sum in (6.9) is also monotonically decreasing with a_u in the interval $0 \leq a_u \leq r_u$. There will be a region around the decision boundary where the approximation will not yield the optimal result (the minimum will be attained for a different set of a_u, thus a different set d will be indicated by (6.9)). This will be illustrated in Section 6.3.1 below.

The meaning of $a_u \geq 0$ is that no client can receive a negative bandwidth, which makes sense. The restriction on a_u not being greater than r_u means that no data flow must ever be allocated more bandwidth than its output buffer can fill up. This is a requirement for the approximation to maintain the monotonicity from the original equation (6.5). However, analyzing this further, we will realize that this upper limit will only be reached when buffers are being completely drained. Therefore the possible non-optimality of the resulting decision vector due to the limits on a_u is not a major issue.

The result (6.10) is still not very useful, since we need to find the d that maximizes the sum $\sum_{u=1}^{U} P_u r_u a_u(d)$, and all the a_u depend on each other through the decision vector d. Changing one element in d will change the value of two elements in the allocation vector a. However, examining the structure of the capacity matrix C, we will see that a change of variables will help us considerably. Since a_u is the per-flow sum over the resource bins of the allocated capacity from C, and $d_s = u, u \in \{1, ..., U\}, s = 1, ..., S$ (see (6.3)) we may as well express the sum (6.10) in the following way:

$$\sum_{u=1}^{U} P_u r_u a_u(d) = \sum_{s=1}^{S} P_{d_s} r_{d_s} C(d_s, s) \tag{6.11}$$

By expressing the sum in this way, it becomes possible to maximize the sum by simply maximizing each term independently. This can be done since for each resource bin s we can choose the $P_u r_u C(u, s)$ that is maximum over the client data flows u, that is

$$\hat{d}_s = \arg\max_u \left(P_u r_u C(u, s)\right), \qquad s = 1, ..., S, \tag{6.12}$$

which is very simple to do. No corresponding algorithmic simplifications seems possible for the original quadratic criterion.

Criteria used by other scheduling algorithms, similar to (6.12), are presented in Section 5.4.2 and [67, 4, 5, 6]. However, they use other, but intuitively almost equivalent, measures for discriminating between clients.

6.2.1 Alternative Derivation 1: Differentiation

The same result as obtained in (6.12) can be derived by differentiation of the cost function J, with respect to the buffer levels r_u. Start from the expression for the quadratic cost function with no allocations ($a_u = 0$)

$$J = \sum_{u=1}^{U} P_u r_u^2. \tag{6.13}$$

Differentiate (6.13) with respect to r_u,

$$\frac{\delta J_{(u)}}{\delta r_u} = 2P_u r_u, \quad , u = 1, ..., U. \tag{6.14}$$

If infinitisimally small changes δr_u of the buffer levels could be realized, they would thus result in

$$\delta J_{(u)} = 2P_u r_u \delta r_u, \quad , u = 1, ..., U. \tag{6.15}$$

For a finite, realizable, variation $-\Delta r_u = a_u$, which represents a decrease in buffer level due to a whole resource bin being assigned to client u, we therefore obtain

$$\Delta J_{(u)} \approx -2P_u r_u \Delta r_u = 2P_u r_u a_u, \quad , u = 1, ..., U. \tag{6.16}$$

Looking at one slot at the time, we would assign the slot to the stream u that would minimize ΔJ, or maximize $-\Delta J$, in the current slot, thereby (approximately) performing the largest possible reduction of J in (6.13).

6.2.2 Alternative Derivation 2: Taylor expansion

Again regard J in (6.13) as a function of the buffer levels $r_u, u = 1, ..., U$. The current buffer levels are denoted r_{u0}, whereas the buffer levels *after* executing the scheduling decision are denoted r_{u1}. We will now make a first order Taylor expansion around $r_u = r_{u0}$, and evaluate it in $r_u = r_{u1}$.

The first order Taylor expansion of $f(x)$ around $x = b$ is given by:

$$f(x) \approx f(b) + f'(x)|_{x=b}(x - b) \tag{6.17}$$

Applying this on our criterion J in (6.13) gives

$$\begin{aligned} J(x) &\approx \sum_{u=1}^{U} P_u \cdot b^2 + 2\sum_{u=1}^{U} P_u \cdot b \cdot (x - b) \\ J(r_{u1}) &\approx \sum_{u=1}^{U} P_u r_{u0}^2 + \sum_{u=1}^{U} 2P_u r_{u0}(r_{u1} - r_{u0}) \\ &= \sum_{u=1}^{U} P_u r_{u0}^2 - \sum_{u=1}^{U} 2P_u r_{u0} a_u. \end{aligned} \tag{6.18}$$

From (6.18) we observe that in order to minimize $J(r_{u1})$ we should maximize

$$\sum_{u=1}^{U} P_u r_{u0} a_u,$$

which brings us back to (6.10), from where we can apply the same reasoning to arrive at (6.12).

6.3 Algorithms based on the Quadratic Criterion

The algorithms described here are based on the quadratic criterion (6.2) presented in Section 6.1, using its linear approximation from (6.12). Namely, we define two algorithms, one with an intermediate update of the buffer level, and one without. Moreover, we present a search based algorithm in Section 6.3.4, that explicitly uses the quadratic criterion (6.2). First, we motivate the development of two alternative algorithms based on the linear approximation.

6.3.1 Motivation of Different Linear Algorithms

Figure 6.1 illustrates how a linear approximation (6.12) of the quadratic criterion (6.2) would affect one term in the linear criterion function as compared to the quadratic criterion.

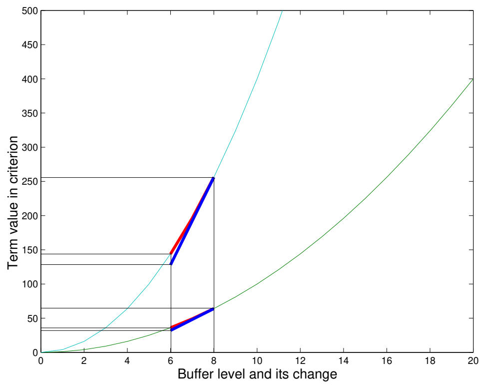


Figure 6.1: Illustration for two buffers with different weighting factors P_u. The linear approximation in each term results in deviations from the exact criterion, but discrimination properties remain for reasonable allocation sizes.

Allocating a single resource bin to a stream based on this linear approximation would result in suboptimal scheduling decisions only in some rare cases where the discriminations between the quadratic and the linear functions yield different decisions.

However, when scheduling a large number of resource bins simultaneously, over few clients, the scheduled decision may lead to a larger deviation from the optimal one if care is not taken (see Figure 6.2). This happens since the amount of resources allocated to each client may become large, thereby making the sum of the linear functions deviate more from the quadratic curvature of the original criterion function. This may be avoided by regularly "calibrating" the linear criterion function with the actual buffer level (see Figure 6.3), making the linear functions obtain their gradient from the original quadratic function.

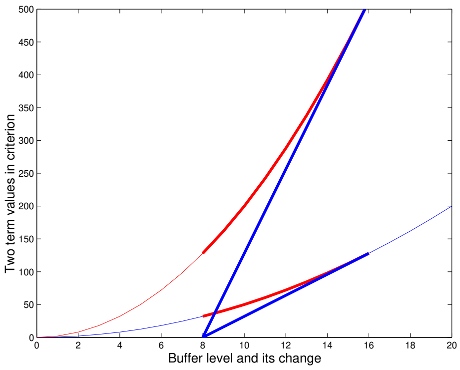


Figure 6.2: If the allocation size becomes large for one client, then the deviation of the linear approximation from the quadratic criterion tends to increase. In this case the candidate allocation was 8 (4+4) for both buffers. This approximation is used in the MAXR algorithm, described later in Section 6.3.2.

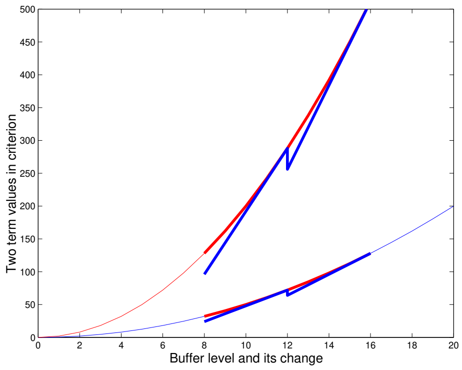


Figure 6.3: The deviation of the linear approximation from the quadratic criterion is kept small if the linear approximation is updated regularly, i.e. the buffer size that dictates the gradient of the linear approximation is updated to use the new gradient. The candidate allocation was 8 (4+4) for both buffers. This approximation is used in the ITER algorithm, described later in Section 6.3.3.

Thus, we see that it may be motivated with two alternative algorithms based on the linear approximation (6.12), in order to allow for a simpler approach (MAXR) when there are relatively many active clients, while we may use the more accurate, but more computationally demanding approximation (ITER), when fewer clients are active.

6.3.2 Non-updated Linear Algorithm (MAXR)

MAXR[4] uses the non-updated linear approximation from Figure 6.2.

1. For each resource bin s and client u, calculate the approximate contribution ΔJ_{us} from (6.12) to reduce the cost function (6.2):

$$\Delta J_{us} = P_u \cdot r_u \cdot C(u, s), \quad s = 1...S, \quad u = 1...U$$

2. Allocate each bin s to the client u with the largest contribution ΔJ_{us} (ties are broken randomly) to reduce the cost function (6.2)

$$d_s = \hat{u} = \arg\max_u \Delta J_{us}, \quad s = 1...S$$

The computational complexity for MAXR is summarized in Table 6.1 in terms of number of required operations to schedule U clients on S resource bins.

Operation	Count	Iterations	Complexity	Total
$P_{ur} = P_u \cdot r_u$	U	1	1	U
$\Delta J_{us} = P_{ur} \cdot C(u, s)$	$U \cdot S$	1	1	$U \cdot S$
$\max_u \Delta J_{us}$	S	1	U	$U \cdot S$
Total				$U + 2US$

Table 6.1: Number of operations for the MAXR algorithm. The column "Count" refers to the number of times the operation occurs in one iteration, "Iterations" refers to the number of iterations until the final result is obtained, and "Complexity" refers to the (approximative) computational complexity of one operation.

6.3.3 Updated Linear Algorithm (ITER)

ITER[5] uses the updated linear approximation from Figure 6.3.

[4] "MAX" comes from the fact that we simply choose the *maximum* element in an array, whereas "R" is appended since we choose *randomly* among the maximum elements when it is not unique.

[5] "ITER" is a suitable name for this algorithm since the method iterates over the resource bins, updating the requirement vector after each resource bin allocation. "R" refers to that ties are broken randomly.

1. For each client u, calculate the approximate contribution to minimize the cost function (6.2), by multiplying each client u's best rate, $\max_{s \in \mathcal{S}} C(u,s)$, with its respective output buffer level r_u and its weighting factor P_u:

$$\Delta J_u = P_u \cdot r_u \cdot \max_{s \in \mathcal{S}_0}(C(u,s)), \qquad u = 1...U,$$

   where

$$\mathcal{S}_0 = \{1, ..., S\}.$$

   The bin index to each client u's best rate in the capacity matrix C is s_u, given by:

$$s_u = \arg\max_{s \in \mathcal{S}_0}(C(u,s)), \qquad s \in \mathcal{S}_0, \qquad u = 1...U$$

2. Find the client $\hat{u}$ that attains the highest ΔJ_u in any bin (ties are broken randomly). Allocate this bin to client $\hat{u}$, and denote it $s_{\hat{u}}$:

$$d_{s_{\hat{u}}} = \hat{u} = \arg\max_u \Delta J_u$$

3. Remove bin $s_{\hat{u}}$ from the competition:

$$\mathcal{S}_n = \mathcal{S}_{n-1} \setminus s_{\hat{u}},$$

   where $\mathcal{A} \setminus a$ denotes that item a is removed from the set $\mathcal{A}$.

4. Update client $\hat{u}$'s output buffer level according to its recent allocation and recalculate its contribution with its next best bin capacity and weighting factor:

$$\Delta J_{\hat{u}} = P_{\hat{u}} \cdot (r_{\hat{u}} - C(d_{s_{\hat{u}}}, s_{\hat{u}})) \cdot \max_{s \in \mathcal{S}_n}(C(\hat{u},s))$$

5. Recalculate the contribution for the other clients $\tilde{u}$ that also had the bin $s_{\hat{u}} = \arg\max_s(C(u,s)) = s_{\tilde{u}}$ as their best candidate:

$$\Delta J_{\tilde{u}} = P_{\tilde{u}} \cdot r_{\tilde{u}} \cdot \max_{s \in \mathcal{S}_n}(C(\tilde{u},s))$$

6. Goto 2 until all bins s have been allocated to a client u.

Operation	Count	Iterations	Complexity	Total
$P_{ur} = P_u \cdot r_u$	U	1	1	U
$C_u = \max_s C(u,s)$	U	1	S	$U \cdot S$
$C_{\hat{u}} = \max_s C_{\hat{u}s}$	μ	S	$S/2$	$\frac{\mu S^2}{2}$
$\Delta J_u = P_{ur} \cdot C_u$	U	S	1	$U \cdot S$
$\max_u \Delta J_u$	1	S	U	$U \cdot S$
$P_{\hat{u}r} = P_{\hat{u}}(r_{\hat{u}} - C_{\hat{u}s})$	1	S	2	$2S$
Total				$U + 3US + \frac{\mu S^2}{2} + 2S$

Table 6.2: Number of operations for the ITER algorithm. Here, $1 \leq \mu \leq U$ equals the number of clients $\tilde{u}$, that also had bin $s_{\hat{u}}$ as their best candidate resource bin.

The difference betwen MAXR and ITER is in the recalculation of the output buffer levels (token bucket levels) in step 4. With this step we come closer to the quadratic function, which can be seen in Section 6.2.2, since we evaluate the approximation of J in a point closer to the point of the Taylor expansion.

The computational complexity for ITER is summarized in Table 6.2 in terms of number of required operations to schedule U clients on S resource bins. It is clear that the additional accuracy of the ITER algorithm comes at an increased computational cost, which is $US + \frac{\mu S^2}{2} + 2S$.

6.3.4 Controlled Steepest Descent (CSD)

CSD[6] utilizes the criterion (6.2) directly for evaluation of the instantaneous cost function. By creating candidate schedules, re-allocating one time-slot at a time, then committing the best candidate re-allocation (the transaction that results in the smallest (6.2)) we hope to find a near-optimal schedule: The algorithm is initialized with the schedule provided by the ITER algorithm, from Section 6.3.3. The ITER algorithm was chosen, since it comes close to the CSD solution, thereby keeping the number of relatively complex CSD steps small. An alternative initial guess could be given by the MAXR algorithm, but that would require a longer search path with the CSD algorithm.

1. Iterate for each resource bin (S resource bins)

[6] "Controlled" refers to the procedure where we evaluate all alternative one-step changes to the candidate solutions, and then *select* the alternative that reduces (6.2) the most. Thus, there is no *gradient* involved, since we explicitly test all solutions that are one step away.

(a) Evaluate the reduction of (6.2), when an assigned resource bin is taken from the assigned client (one re-evaluation of one term in (6.2)), and given to each of the other clients ($U-1$ re-evaluations of one term in (6.2))

2. After all the possible one-step transactions have been evaluated, the one resulting in the the smallest (6.2) is executed.

3. Iterate the procedure, until no decrease in (6.2) is accomplished.

The computational complexity for the CSD algorithm is presented in Table 6.3 in terms of the number of required operations in order to perform one step in the CSD algorithm. The number of steps until CSD converges

Operation	Count	Iterations	Complexity	Total
$J_{u\hat{s}} = J_{us} - (r_u - a_u)^2 + (r_u - (a_u - C(u, \hat{s})))^2$	1	S	7	$7S$
$J_{j\hat{s}} = J_{u\hat{s}} - (r_j - a_j)^2 + (r_j - (a_j + C(j, \hat{s})))^2$	$U-1$	S	7	$7S \cdot (U-1)$
$\max_j \max_{\hat{s}} J_{j\hat{s}}$	1	1	$U \cdot S$	$U \cdot S$
Total				$8US$

Table 6.3: Number of operations for the CSD algorithm *in each step*, after having run the ITER algorithm to find the initial point. The number of required steps for convergence depends on the intial point from where the CSD starts improving the schedule. The number of steps is bounded by US, but has not exceeded S in any simulation, when starting from the soultion found by the ITER algorithm.

depends on the initial point, but it is bounded by US (including evaluation of the initial point), thus, the worst case for its complexity is $8U^2S^2$. In none of the cases tested in the simulations, did the number of steps exceed S, suggesting that a more reasonable total complexity estimate is $\leq 8US^2$

6.3.5 Summary of Computational Complexity

In Figure 6.4 we summarize the findings in Tables 6.1 to 6.3, by assigning values to the variables U and S and calculating $10 \cdot \log_{10}(f(U, S))$, where

$$f(U,S) = \begin{cases} U + 2US & \text{for MAXR} \\ U + 3US + S^2 + 2S & \text{for ITER} \\ 8US^2 & \text{for CSD} \end{cases} \tag{6.19}$$

Thus, we have assumed that the CSD algorithm takes S steps to converge, and also disregarded the complexity required for finding the initial schedule. Furthermore, we used $\mu = 2$ for the ITER algorithm. We find that the

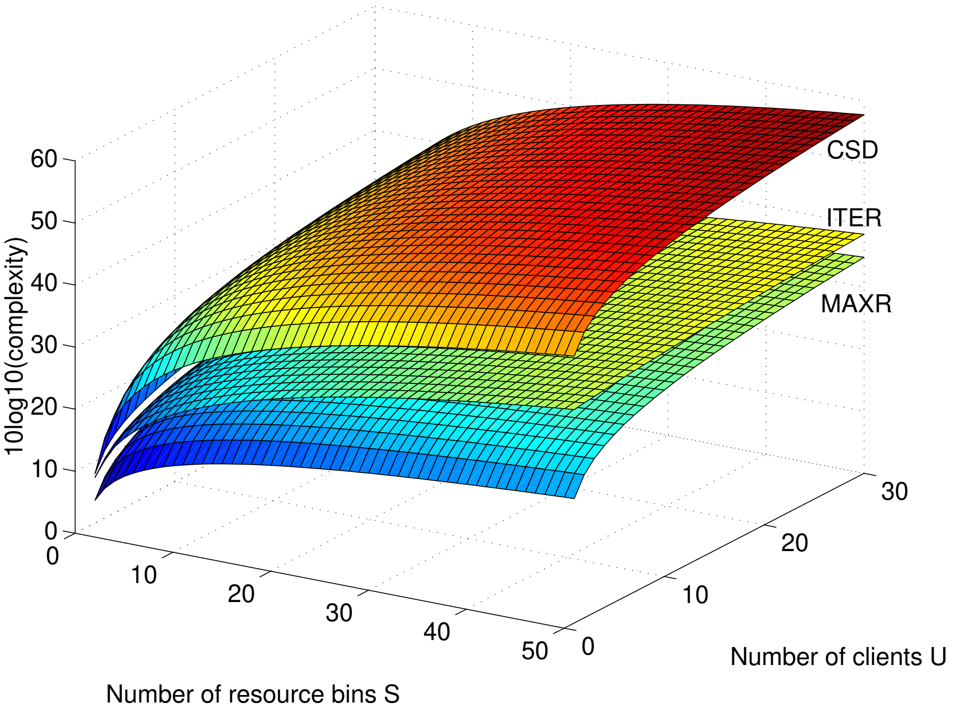


Figure 6.4: Comparison of the computational complexity for the three outlined algorithms. The complexity is represented as $10\log_{10}(f(U,S))$, where $f(U,S)$ is given in Equation (6.19).

MAXR is the least complex of all three in all cases. Furthermore, the difference between ITER and MAXR depends more on S than on U, which is a nice feature since it is more probable that the number of clients will vary, than the number of available resource bins. It is also clear that the CSD algorithm is several orders of magnitude more computationally demanding than the other two.

Example 6.5: A 100 MIPS processor for scheduling

Using a 100 MIPS processor and requiring a new schedule every 0.667 ms, we may allow a computational complexity of 48 on the z-axis of Figure 6.4. This qualifies CSD as a possible algorithm for smaller schedules. For $S = 25$, CSD would be able to handle $U = 13$ clients. However, using the MAXR algorithm, that in the largest considered schedule ($S = 50$ and $U = 30$) has a computational complexity of 35, the margin up to 48 is 13, therefore allowing one 100 MIPS processor to handle $2^{13/3} > 20$ parallel such schedules. The corresponding value for the ITER algorithm is $2^{(48-39)/3} = 8$ parallel schedules.

6.3.6 Deviations from Optimum by the Approximations

The linear approximation provides an attractive simplification of the optimization task, though it may not always result in the same decision as the original criterion. In Section 6.3.1 we motivated the use of two different algorithms based on the linear approximation (6.12). In this section we will compare their scheduling performance and compare them to the optimal schedule according to (6.2), found by an exhaustive search.

In the following we have explored a large number of cases for a rather small scheduling problem: Two clients to be scheduled for three resource bins. In Figure 6.5 below we have compared the optimal solutions, with the ones found by the two different linear approximations, MAXR from Section 6.3.2 (left part of Figure 6.5), and ITER from Section 6.3.3 (right part of Figure 6.5). In Figure 6.6 we have compared the optimal solutions with the ones found by the serach based method, CSD from Section 6.3.4.

The calculations have been done as follows:

- We have a scheduling problem with two clients $u \in \{1, 2\}$ and three resource bins $s = 1, 2, 3$. The two clients will be scheduled to use the three bins according to their requirements r_u and their bin capacities $C(u, s)$.

- The buffer levels r_u are represented in the figures, and they can most easily be recognized in Figure 6.7, where we have a blow-up of a part of Figure 6.5. In Figure 6.7, the r_u are represented along both axes, one for each client u, on all four plots:

  - r_1 along the horizontal axis, ranging from $r_1 = 1$ to $r_1 = 30$.
  - r_2 along the vertical axis, ranging from $r_2 = 1$ to $r_2 = 30$.

The buffer level values are looped through, from $r_u = 1$ to $r_u = 30$, in order to evaluate the scheduling method for all these cases.

- The bin capacities for the two clients in the three resource bins are also partly represented in the figures.
  - For client number 1, the bin capacity is 4, in all three resource bins, which is not represented in the figures: $C(1,1) = C(1,2) = C(1,3) = 4$
  - For client number 2, the bin capacity in resource bin 3 is always 7, which is also not represented in the figures: $C(2,3) = 7$
  - However, for client number 2, resource bins 1 and 2 are represented in Figure 6.5: $C(2,1)$ along the horizontal axis, and $C(2,2)$ along the vertical axis, in all four plots.

  The bin capacity values are looped through, from $C(2,1) = 0$ to $C(2,1) = 7$, and from $C(2,2) = 0$ to $C(2,2) = 7$, for each combination of r_1 and r_2 above.

- The colours in the graphs represent the difference between the optimal solution using (6.2), and its approximations. The values presented have been calculated for each point in the graphs by:
  1. Computing the optimal solution, using an exhaustive search with the quadratic criterion (6.2).
  2. Finding the approximate solution using one of the suggested algorithms.
  3. Finding the difference in criterion value by:
     (a) Evaluating the optimal solution using (6.2).
     (b) Evaluating the approximate solution using (6.2).
     (c) Computing the difference between them and present the value in the upper graphs.

     and
  4. Finding the difference in the decision vector by:
     (a) Computing the decision resulting from the optimal solution and the decision resulting from the approximate solution.
     (b) Counting the number of resource bins in which they disagree and present the value in the lower graphs.

- The upper left graph in Figure 6.5 shows the comparison of the exact *criterion value* with the non-updated approximation.
- The upper right graph in Figure 6.5 shows the comparison of the exact *criterion value* with the updated approximation.
- The lower left graph in Figure 6.5 shows the comparison of the *decision vector* according to the exact criterion with the non-updated approximation.
- The lower right graph in Figure 6.5 shows the comparison of the *decision vector* according to the exact criterion with the updated approximation.

In short, white is good and black is bad. The darker the colour, the bigger the difference. Thus, a dark spot in the upper left graph in Figure 6.5, means that there is a big difference in criterion value between the exact solution (minimal J) and the solution found with the non-updated approximation. Furthermore, the position of the dark spot defines the values of r_1, r_2, $C(2,1)$, and $C(2,2)$, yielding that solution. (The other values, $C(1,1)$, $C(1,2)$, $C(1,3)$, and $C(2,3)$, are constant and given above.)

Observation 6.1: *ITER is more accurate than MAXR*

Figure 6.5 indicates that the updated linear approximation results in a decision more in line with the quadratic criterion, than the non-updated.

■

Observation 6.2: 26.0% *of CSD scheduling decisions disagree with optimal decisions*

The CSD algorithm disagrees from optimum in only 26.0% of the scheduling decisions. However, the criterion value difference is zero for most of the cases, so the decisions are still optimal (the optimal solutions are not unique). The corresponding number is 36.3% for the ITER algorithm, and 38.1% for the MAXR algorithm.

■

To sum up the analysis of the performance of the linear approximations, we conclude that the ITER and CSD algorithms clearly show much better performance than the MAXR algorithm. This is particularly clear when comparing the differences between the resulting criterion values from the

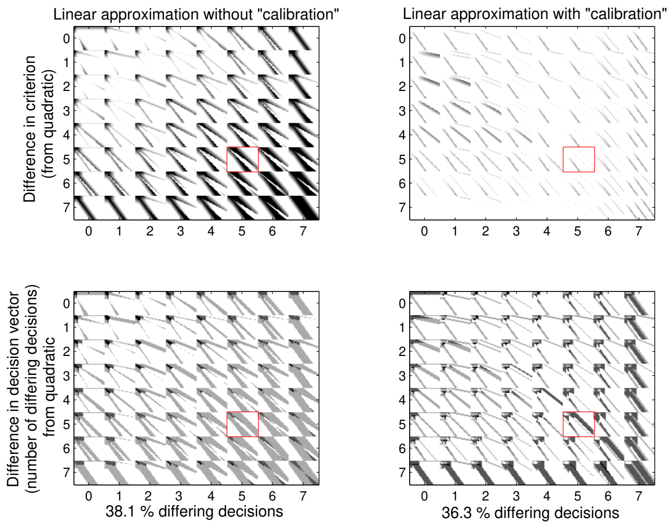


Figure 6.5: The two top graphs illustrate how much the quadratic criterion values differ from the optimal when using the two linear approximations. The darker the color, the greater the difference (white means identical). The left one is for the case when the buffer level is not updated between each slot allocation (the MAXR algorithm), while the right one enjoys an update of the buffer level (the gradient in the linear criterion) between each slot allocation (the ITER algorithm). The two lower graphs show the corresponding difference in the decision (which stream is allocated which slot) between the quadratic criterion and the two respective linear approximation versions. White color means identical decisions (zero difference), while the darker colors indicate difference in one, two, or three slots (light gray, dark gray, and black) respectively. The position for each colored pixel indicates the channel conditions and the buffer level status as follows: *Bin capacities:* The labels on the x- and y-axes indicate the possible transmission rate for stream 2 in slot 1 and 2 respectively. The transmission rate for stream 2 in slot 3 is always 7 (maximum). The transmission rates for stream 1 are all 4 for all three slots. *Buffer level:* Within each channel status field, the two streams' buffers initial levels are varied from 1 to 30 along the x- and y-axes. The small squares show the position of the blow-ups in Figure 6.7

algortihms. MAXR shows large deviations from the optimum in many cases, and they become larger as the difference in bin capacities between the two users, grows. With this in mind, we will prefer to utilize ITER and CSD. In the cases where we *do* use MAXR, as we will in Chapter 8, we will have to pay attention to its inferior performance, in the quadratic criterion sense.

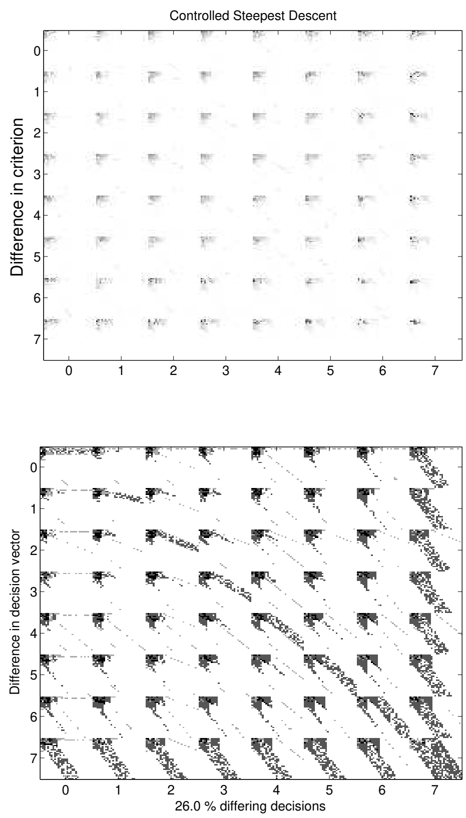


Figure 6.6: For comparison, also the resulting differences for the Controlled Steepest Descent algorithm are presented. Clearly, it is better than MAXR and ITER at approximating the quadratic criterion, which is not surprising, since CSD starts to search for improved schedules from the ITER solution.

6.4 Summary

We have presented a general criterion (6.1) for resource scheduling, that takes service requirements and price model parameters, as well as channel properties, into account. We chose to elaborate on a special case of (6.1), namely the quadratic criterion (6.2). Based on this criterion, we developed a linear approximation that considerably simplifies the (sub-) optimization of the resource utilization schedule. We proposed two algorithms of different complexity and performance, based on this approximation, making them adequate for on-line usage. Furthermore, we also proposed a more complex, search-based algorithm, that enjoys better performance, since it utilizes the

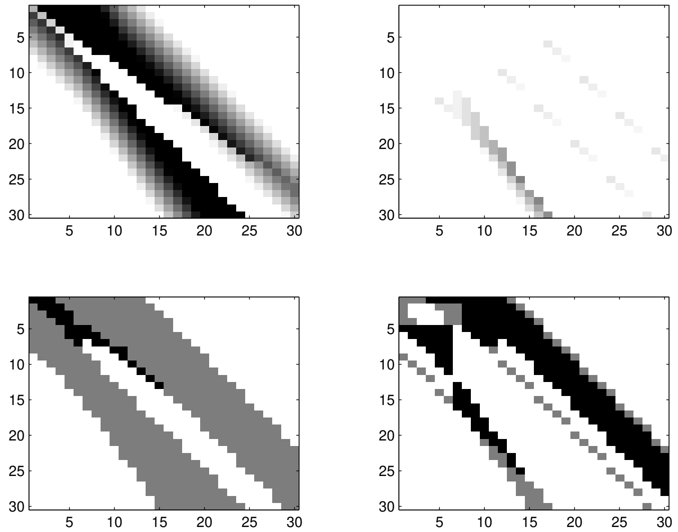


Figure 6.7: Local blow-ups of the red-marked regions in the graphs in Figure 6.5, respectively. The axes have values between 1 and 30, which indicate the initial buffer levels for the two streams, before scheduling. The darkness of each pixel indicates the difference in the quadratic criterion, from the optimal, using the two different linear approximations. For the two top images, the darker the color, the greater the difference. In the two lower images, white color means identical decisions (zero difference), while the darker colors indicate difference in one, two, or three slots (light gray, dark gray, and black) respectively.

original quadratic criterion.

We found that the ITER algorithm possesses both acceptable performance and low-enough complexity, to be a candidate for implementation in a real multi-user link scheduler. In the case study, in Chapter 9, we will evaluate the ITER algorithm in more realistic and complex cases than the ones pursued in this chapter.

Chapter 7

Probability based Scheduling Criteria

The quadratic criterion and its approximations are examples of criteria that can be used in order to obtain a relevant discrimination between different data flows for the scheduler. Its idea is to select channel resources to serve the buffers, where they make the most benefit in a quadratic sense. The quadratic criterion has some attractive advantages, such as maintaining relative buffer sizes, or delays, between different flows, according to their service requirements. However, sometimes the relative nature of this approach may not suffice. There are services that require low jitter, that is, small variations in the experienced delay. For these classes, the quadratic criterion is not well suited, since it does not impose any absolute requirements on the buffer levels. [1]

In this chapter we will elaborate on criteria that involve probabilities. More specifically we will try to explicitly minimize the probability of failing to meet the service requirements, while maximizing the probability of utilizing the channel resources as efficiently as possible, all in order to maximize the revenue for the operator.

A different way to regard the scheduling problem, than the previously discussed quadratic criteron, is by studying each individual client, its demands, and its channel resources separately:

[1] Note that we discuss in terms of buffer levels, since we assume that the admission control maintains rather constant, or at least controlled, input rates into the output buffers. This is to avoid the necessity of a timer that keeps track of the time each data packet has spent in the buffer.

- The channels that the different streams have to their potential disposal are characterized by certain statistical properties, such as a slowly varying average SNR, a fast variability around this average in time and frequency, and the magnitude of this varability. These properties may be estimated and utilized in scheduling. Based on historical estimated or predicted data, we can estimate a statistical distribution that describes the average SNR and its variability. By comparing each predicted channel SNR to the estimated distribution, we can obtain a probability for obtaining a better SNR value in the near future. If this probability is low, we should take the opportunity to use the current resource.

- We may regard the buffered data in a similar way. If we assume that queued data is served at a constant rate, then the buffer level can be deterministically mapped to a queueing delay. However, the service rate will vary due to the decisions made by the scheduler. Still, we may be able to create a statistical distribution of the service rate. Using this statistical distribution, we can map the current buffer level to a probability of the last packet being served timely. If this probability is low, then we should try to increase it by properly serving that queue.

Given that we have the two probability distributions mentioned above, can we create criteria and scheduling strategies that

- maximize the probability of serving a critical queue with a good channel resource?

- minimize the probability of breaching a delay requirement?

- do one or the other jointly for all clients?

This chapter presents algorithms that have been constructed with these aims in mind.

Remark 7.1: *Throughput requirements are controlled by SLC*
It is worth noting that we will let the scheduler handle delay requirements, whereas the throughput requirements will be handled by the service level controller (see Section 3.3), through the admitted service rates. The scheduler will then make decisions that efficiently utilize the channel resources, while maintaining reasonable buffer levels. [2]

■

[2] Naturally, the SLC and the scheduler need to *cooperate*, since SLC must ensure that the throughput requirements are feasible with the available resources.

7.1 Probability of Service Failure

If future[3] service (data transmission) rates would be known, then we could determine the time each data packet would spend in the queue, and thereby have control of the queueing delay. Unfortunately, the transmission rates depend on future capacity fluctuations, service demand variations, and scheduler decisions. However, we may look at historical data and, through statistical models, find measures of the urgency of serving a particular queue u.

The concept of service failure is developed below, where we consider the delay requirements for different clients. However, we may expand this reasoning to include other service parameters, such as jitter, which we will do in the following section.

7.1.1 Delay Requirements

We will need to define the *transmission rate* and the *required transmission rate* in order to proceed with the service failure probabilities.

Definition 7.1: *Transmission rate* $R_u(t)$

The *transmission rate* $R_u(t)$ for queue u at time t, is the amount of data transmitted from queue u, during scheduling round t.

■

In most cases, $R_u(t)$ equals the components in the allocation vector, a_u, of scheduling round t. However, when a buffer runs empty ($a_u \geq r_u$), then $a_u \geq R_u(t)$, since the allocation a_u could not be entirely utilized. Knowing the transmission rate $R_u(t)$ for all $t_0 \leq t \leq t_1$, we may introduce the following definition:

Definition 7.2: *Achieved transmission* $B_u(t_0, t_1)$

$$B_u(t_0, t_1) = \sum_{t=t_0}^{t_1} R_u(t) \tag{7.1}$$

is the *achieved transmission*, or, amount of data (number of bytes or tokens) drained from queue u in the time interval

$$t_0 \leq t \leq t_1.$$

■

[3] We here refer to horizons of tens to hundreds of milliseconds, thus out of the scope for a scheduler operating in a wireless communication system.

We may use the inverse of relation (7.1) in order to estimate the *required transmission rate* $\tilde{R}_u$ for queue u, in order to *achieve a transmission* of $B_u(t_0, t_1)$ bytes, or tokens, within the time interval $t_0 < t < t_1$.

Definition 7.3: *Required transmission rate* $\tilde{R}_u$*, and time to live* T_u

The *required transmission rate* is the constant transmission rate, $\tilde{R}_u$, required for draining queue u, with level B_u, in a time interval of length $T_u = t_1 - t_0$. It is given by

$$\tilde{R}_u = \frac{B_u}{T_u}. \tag{7.2}$$

Here, T_u is said to be the *time to live* for the queueing data at queue level B_u.

■

Remark 7.2: *Target delay of recently inserted packet*

For a recently inserted data packet, T_u in Definition 7.3 equals the *target delay* for the service class to which queue u belongs.

■

We may collect statistics of $R_u(t)$ (Definition 7.1 above) and find a probability distribution $f_{\mathcal{R}_u}(t, r)$ for the random variable $\mathcal{R}_u$, representing transmission rate (that may be time-varying), in order to calculate the probability,

$$P(\mathcal{R}_u \leq x) = F_{\mathcal{R}_u}(t, x) = \int_0^x f_{\mathcal{R}_u}(t, r)dr, \tag{7.3}$$

of obtaining transmission rate $\mathcal{R}_u = x$ or less.

Definition 7.4: *(Individual) failure probability* P_u^F *and success probability* P_u^S

The *failure probability* for client u, is

$$P_u^F = P(\mathcal{R}_u \leq \tilde{R}_u) = F_{\mathcal{R}_u}(t, \tilde{R}_u) = \int_0^{\tilde{R}_u} f_{\mathcal{R}_u}(t, r)dr, \tag{7.4}$$

The *success probability* is the complement of the failure probability:

$$P_u^S = 1 - P_u^F \tag{7.5}$$

■

Remark 7.3: *Failure probability and target delay*

Equation (7.4) gives the probability of *not* being able to serve a data packet of client u within its time to live, T_u.

■

It is understood in the equations (7.3)-(7.5) that the probabilities and the distributions may be time-varying.

Observation 7.1: *Estimation of service rate probability distribution*

The CDF $F_{\mathcal{R}_u}(t, x)$ in (7.3) can be obtained from a sliding window of previously achieved transmission rates

$$R_u(\tau), \qquad t - T \leq \tau \leq t.$$

■

A cumulative distribution function (CDF) may be represented in a multitude of forms. One way is to assume some distibution function, and then estimate its parameters. Another way is by histogram methods. A third way, which we will expound on in Section 7.4.2, uses a sorted list of historical data. Bayesian methods for constructing histograms that represent PDF:s and CDF:s, using Laplace rule of succession, have recently been developed in [48].

Before we proceed, some reflections on scheduling delay sensitive clients, are in place (see also Figure 7.1):

- The required transmission rate, after a scheduling decision, is desired to be as small as possible for each client, in order to minimize the failure probability (7.4).

- Each resource allocation contributes to reducing the remaining required rate, and thus to reducing the failure probability.

For best effort services, that may not have any explicit delay requirements, their service failure probabilities can be set to a client-dependent, fixed, low value. That value reflects the revenue that is generated by serving them, as compared to serving clients with delay requirements, and other best effort clients. This value can be seen as a threshold meaning that failure probabilities below that value are not urgent. For other data services, it must be noted that data transmission services, for carrying transport protocols such as TCP, are sometimes named "best effort", even though they implicitly assume some limited delay for the transmission to be meaningful. These types of services *should* specify explicit delay requirements, and be priced

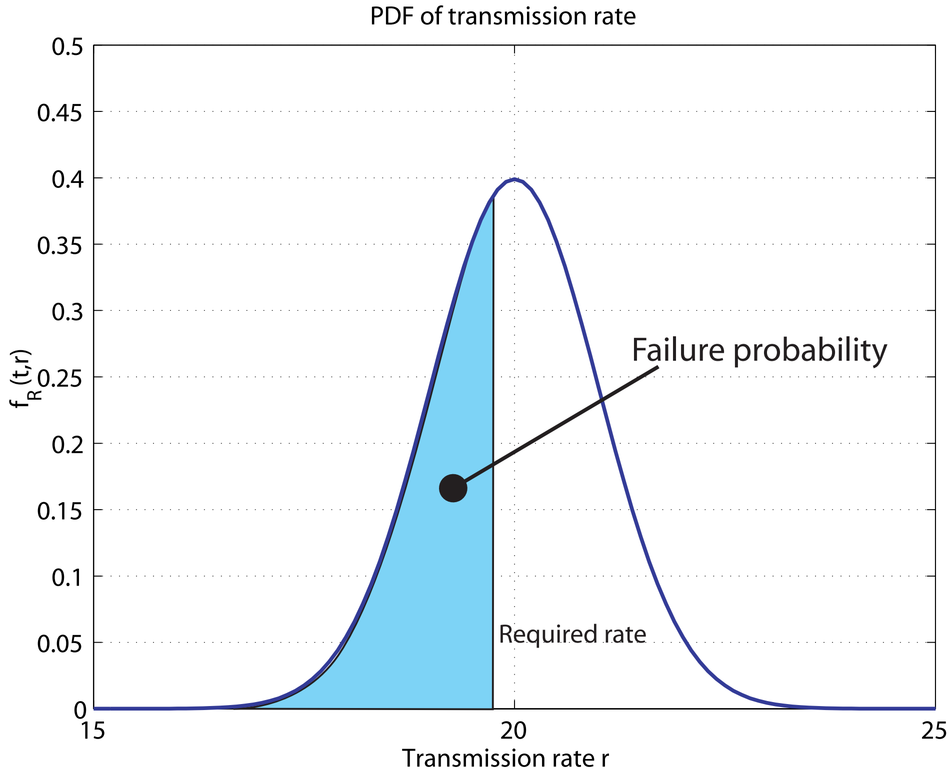


Figure 7.1: The failure probability for a delay constrained client, given its transmission rate PDF. The lower the required rate, the lower the failure probability. A resource allocation reduces the remaining required rate.

accordingly, in the SLA:s, in order to be handled correctly by the scheduler and the admission control.

Next we define the *overall* failure probability.

Definition 7.5: *Overall failure probability P_F and success probability P_S*

With *overall failure probability*, P_F, is meant the probability for failing with at least one client, which in turn is the complement to the probability of not failing with any client:

$$\begin{aligned} P_F &= P(\text{Failure}) \\ &= 1 - P(\text{Not failing with any client}) \\ &= 1 - P(\text{Success } 1 \cap \text{Success } 2 \cap \ldots \cap \text{Success U}) \end{aligned} \quad (7.6)$$

The *overall success probability* P_S is the complement of the overall failure probability P_F,

$$P_S = 1 - P_F. \tag{7.7}$$

■

The maximization of the overall success probability would be a combinatorial problem that does not have a simple solution, since it includes the channel properties (transmission capacities) implicitly. In this sense it is similar to the original quadratic criterion (6.2) of Section 6.1. However, we shall aim for approximations that perform well enough. The first step we have already prepared for: We are *not* considering a maximization of the overall success probability P_S. We will here instead focus on the individual failure probabilities P_u^F, as indicators of the *urgency* of serving a particular client u.

Remark 7.4: *Queue stability not guaranteed*

Since the function of the buffer level, in this case the failure probability, $0 \leq P_u^F \leq 1$, is a bounded function, the resulting scheduling strategies will not enjoy the stability properties outlined in Appendix B.1. Therefore, it is important for the service level controller to monitor the buffer levels and adapt the admitted rates accordingly.

■

The second step includes looking at how the channel properties can be taken into account explicitly in the allocation, since we do not want to waste low quality resources on urgent clients, if we expect the resource quality to improve for those clients in the near future.

Before we proceed to discuss resources, we indicate how to include also jitter requirements into the service failure probability.

7.1.2 Jitter Requirements

We introduced the delay requirements, and the probability to fulfill them, by defining the required rate $\tilde{R}_u$ in terms of the buffer level B_u and the target delay T_u. Let us in the same way introduce the *maximum* and *minimum* required rates $\overline{\tilde{R}}_u$ and $\underline{\tilde{R}}_u$, respectively:

Definition 7.6: *Maximum and minimum required rates, $\overline{\tilde{R}}_u$ and $\underline{\tilde{R}}_u$*

Let T_u denote the target delay for client u. Furthermore, let $t_u > 0$ denote the acceptable jitter for client u, so that

$$T_u - t_u/2 < T_u < T_u + t_u/2.$$

For a given output buffer level, B_u, define the *maximum required rate* as

$$\overline{\tilde{R}}_u = \frac{B_u}{T_u - t_u/2}, \tag{7.8}$$

and the *minimum required rate* as

$$\underline{\tilde{R}}_u = \frac{B_u}{T_u + t_u/2}. \tag{7.9}$$

■

We may still collect statistics of $R_u(t)$ (Definition 7.1 above) and find a probability distribution $f_{\mathcal{R}_u}(t, r)$ (that may be time-varying), in order to calculate the probability,

$$P(x \leq \mathcal{R}_u \leq y) = \int_x^y f_{\mathcal{R}_u}(t, r)dr, \tag{7.10}$$

of obtaining transmission rate $x \leq \mathcal{R}_u \leq y$. The *success probability* for client u in terms of these new service requirements (limited jitter), is obtained by setting x in (7.10) equal to $\underline{\tilde{R}}_u$, and $y = \overline{\tilde{R}}_u$ from Definition 7.6.

$$P_u^{S_j} = P(\underline{\tilde{R}}_u \leq \mathcal{R}_u \leq \overline{\tilde{R}}_u) = \int_{\underline{\tilde{R}}_u}^{\overline{\tilde{R}}_u} f_{\mathcal{R}_u}(t, r)dr, \tag{7.11}$$

In Figure 7.2 we illustrate two interesting scheduling cases involving jitter. By studying (7.11) and Figure 7.2 we observe that there may be a threshold where additional resource allocations to a client actually *reduces* its success probability. The success probability, in the jitter case, is thus not monotonically increasing with the allocated resources, as it is for the delay case.

Note that scheduling resources to a client with success probability $P_u^{S_j}$ as depicted in the left part of Figure 7.2 will increase $P_u^{S_j}$ further. However, if a client's $P_u^{S_j}$ is as in the right part of Figure 7.2, then no additional resources should be scheduled to this client, since they would only worsen the situation by reducing the service success probability even more. As resources are allocated to a client, its required rate will be reduced, since its remaining buffer level B_u, in Definition 7.6, is reduced. However, if the client is not served for a while, then the buffer level B_u will grow and $P_u^{S_j}$ will grow again.

Remark 7.5: *Probability based scheduling shapes the PDF:s*

By scheduling according to the probability densities based on statistics from previous transmissions, we shape the probability densities for future

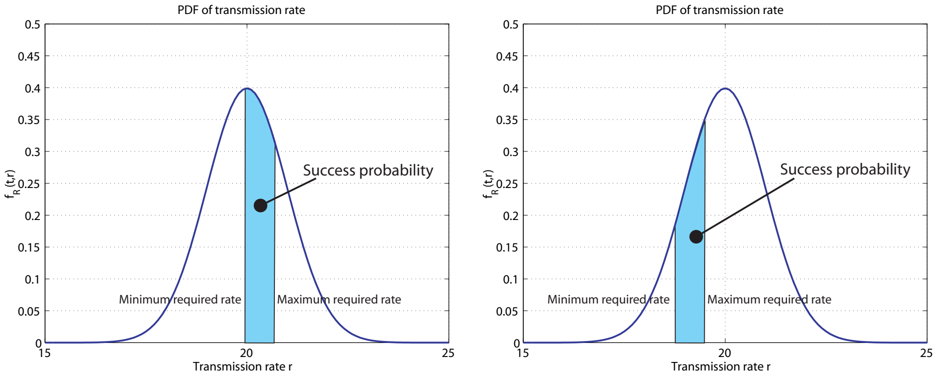


Figure 7.2: The service success probability for a jitter constrained client. To the left, the success probability is high, since it is probable that the achieved rate will be within the limits imposed by the jitter requirement. To the right, the success probability is lower, since it is probable that the achieved rate will be higher.

scheduling. Disregarding the influences from the channel constraints, past successful scheduling will lead to higher future success probabilities (narrower rate distributions).

■

Remark 7.6: *Pricing for jitter requirements*

Since the jitter requirements pose additional service constraints on the resource allocation, service pricing should reflect this by increasing the price as the jitter requirements narrow.

■

7.2 Probability of a Good Resource

In this section, we will see how we can take the statistical properties of the channel into account, explicitly, in the scheduling problem. For the channel conditions, the question we want a suitable probability distribution to answer is:

- Given a current prediction of the channel quality, is it probable that the channel will serve as a better resource later?

This question is central to the scheduling problem. If the distribution function says "no", then we should take the opportunity to exploit the current channel condition, since, according to the statistics, it will rarely get better. If, on the other hand, the distribution function says "yes", we should wait and hope for a better channel quality. Let $\mathcal{C}_u$ be a random variable representing the bin capacity of the channel for client u. We can now express the probability of a good resource.

Definition 7.7: *Probability of a good resource* P_u^C

Given the bin capacity, $C(u,s)$, for client u in resource bin s, we can calculate the *probability of* $C(u,s)$ *being a good resource* P_u^C,

$$\begin{aligned} P_u^C &= P(\mathcal{C}_u < C(u,s)) \\ &= F_{\mathcal{C}_u}(t, C(u,s)) \\ &= \int_{-\infty}^{C(u,s)} f_{\mathcal{C}_u}(t,c)dc, \end{aligned} \tag{7.12}$$

from the distribution $F_{\mathcal{C}_u}(t, \mathcal{C}_u)$.

■

The probability distributions in (7.12) are obtained, for each client u, from past *bin capacities*, $C(u,s)$ (see Definition 4.1 on page 58), where u is an index over the admitted streams, and s over the existing resource bins. Then, for each $C(u,s)$, we can through the capacity cumulative distribution function (CDF), $P(\mathcal{C}_u < C(u,s))$, obtain the probability for it being a good resource bin for client u.

Maximization of (7.12) is simple. It can be performed on a bin-by-bin basis, allocating each bin to the client that has the highest value of (7.12).

If we had only the probability distribution in (7.12), for the purpose of scheduling resources, then we should try to utilize the channel while it is good, so we should select the resource bins that have high probability of being good. However, such a strategy will not give any promises on the performance of the queueing system:

- It will on average allocate an equal number of resource bins to the clients, achieving some kind of proportional fairness among the clients (see [62, 18] and Section 5.3.2).
- It will also result in more spectrally efficient scheduling decisions than the service failure probability minimization from Section 7.1, since it is not constrained by the service requirements.

- However, since it does not take the service requirements into account, it cannot be used for scheduling streams with specific service requirements.

Since we wish to find a scheduling strategy that maximizes the overall success probability, we have to take the service requirements into account. The question we have to answer is how the channel capacity probability and the individual service failure probability can be considered simultaneously. This will be discussed in the following section.

7.3 Combining the Probabilities

The bin capacity statistics and the transmission rate statistics most certainly contain correlations. However, we choose a simpler approach by deliberately not considering the joint statistics, since this would otherwise result in a combinatorial optimization problem, as mentioned in Section 7.1. Thus, taking the role as a scheduler, we will look at the separate statistics, by considering one resource bin and one client, with its probability P_u^C of being good, and its probability P_u^F of failing to fulfill the service requirement.

Table 7.1 summarizes four rules for the appropriate qualitative properties of scheduling rules that are based on both P_u^F and P_u^C.

	P_u^C large	P_u^C small
P_u^F large	Allocate!	Not spectrally efficient
P_u^F small	Not urgent	Don't allocate

Table 7.1: Rule of thumb for allocating resources when P_u^F and P_u^C are considered jointly, for clients u without jitter requirements.

Table 7.1 should be interpreted as a rule of thumb. For example, it states that if we have a small probability for a better channel capacity than the one currently at test (P_u^C large), and a large probability of a service failure for client queue u (P_u^F large), then we should take the opportunity to allocate the currently good channel to serve the urgent client.

Remark 7.7: *Jitter requirements complicate allocation*

When serving clients with strict jitter requirements, the rules in Table 7.1 are not valid. The basic algorithm will be modified to take jitter requirements into account by making a post-allocation check in order to control that the jitter success probability $P_u^{S_j}$ has not been reduced by the allocation, c.f. Figure 7.2.

■

In order for the scheduler to make an efficient decision (in terms of revenue generation), it should take into account information about both the buffer status and the channel state, when deciding which client should transmit in what bin. One way of doing this, is to multiply the two probabilities, $P_u^F \cdot P_u^C$, for each client and each transmission bin. If we regard the transmission rate $R_u(t)$ and the bin capacity $C(u,s)$ as stochastic variables, $\mathcal{R}_u$ and $\mathcal{C}_u$ respectively, then the maximization of the product $P_u^F \cdot P_u^C$ may be interpreted as *maximizing the probability of serving the most urgent client by assigning the best available resource.*

In the next section we will present two algorithms that are based on the maximization of the product $P_u^F \cdot P_u^C$. In other words, they strive to maximize the probability of assigning the best available resource to serve the most urgent client.

7.4 Probabilistic Criterion Algorithms

The basis for a probabilistic criterion algorithm is a model, or a cumulative distribution function (CDF), describing the involved discrimination parameters, $\mathcal{C}_u$ and $\mathcal{R}_u$. We will present two such alternatives. The *CDF based* algorithm is given without specifying the CDF adjustment procedure, whereas for the *score based* algorithm, we concretize how to adjust the CDF:s.

7.4.1 The CDF Based Scheduling Algorithm (CBS)

The algorithm utilizes the achieved rate distribution and the bin capacity distribution for the different clients, in order to calculate the failure probability, P_u^F, and the probability for a resource bin being good, P_u^C. In order to discriminate between the possible schedules, we allocate the resource bins to the client that maximizes the product between the two probabilities, given its current rate requirement and bin capacity. Moreover, the algorithm utilizes a similar update procedure as that of the ITER algorithm (see Section 6.3.3), by updating the buffer levels as the allocations are decided.

The CBS Algorithm

1. Update the bin capacity CDF for each client with the new predicted values,
$$F_{\mathcal{C}_u}(t|t) = g(F_{\mathcal{C}_u}(t|t-1), C(u,s)),$$
where $g(f,x)$ is a recursive update of the function f, using the new values x.

2. Find each client's best resource bin, in terms of the bin capacity, $C(u,s)$
$$C_u = \max_{s \in \mathcal{S}_0} C(u,s), \quad u = 1,...,U$$
and store their respective resource bin indices
$$s_u = \arg\max_{s \in \mathcal{S}_0} C(s,u), \quad u = 1,...,U,$$
where $\mathcal{S}_0$ is the set of all resource bin indices.

3. Calculate each client's probability of $C(u,s_u)$ being a good resource,
$$J_{\mathcal{C}}(u) = P_u^C = F_{\mathcal{C}_u}(t|t; C(u,s_u)), \quad u = 1,...,U, \tag{7.13}$$

4. Calculate the failure probability for each client (c.f. (7.4)),
$$J_{\mathcal{R}}(u) = P_u^F = F_{\mathcal{R}_u}(\tilde{R}_u), \quad u = 1...U \tag{7.14}$$

5. Multiply the two factors, (7.13) and (7.14), and find the client that obtains the maximum value:
$$\hat{u} = \arg\max_u J_{\mathcal{C}}(u) \cdot J_{\mathcal{R}}(u) \tag{7.15}$$

6. Decide to allocate bin $s_{\hat{u}}$ to client $\hat{u}$:
$$d(s_{\hat{u}}) = \hat{u}$$

7. Remove slot $s_{\hat{u}}$ from the competition:
$$\mathcal{S}_n = \mathcal{S}_{n-1} \setminus s_{\hat{u}}$$

8. Update the output buffer level of session $\hat{u}$ according to its recent allocation and recalculate its failure probability:
$$J_{\mathcal{R}}(\hat{u}) = F_{\mathcal{R}_u}(\tilde{R}_{\hat{u}}) = F_{\mathcal{R}_u}(\frac{B_{\hat{u}} - C(\hat{u}, s_{\hat{u}})}{T_{\hat{u}}})$$

9. Recalculate the probability of a good resource $J_{\mathcal{C}}(u) = P_u^C$ for all clients $\tilde{u}$ that had resource bin $s_{\hat{u}} = \arg\max_s(C(u,s)) = s_{\tilde{u}}$ as their best candidate, since bin $s_{\hat{u}}$ is now unavailable:

$$J_{\mathcal{C}}(\tilde{u}) = \max_{s \in \mathcal{S}_n}(F_{\mathcal{C}_{\tilde{u}}}(t|t; C(\tilde{u}, s))$$

10. Goto 5 until all slots s have been allocated to a stream u.

11. Finally update the transmission rate CDF $F_{\mathcal{R}_u}(r)$ with each client's achieved transmission rate,

$$F_{\mathcal{R}_u}(t+1|t; r) = g(F_{\mathcal{R}_u}(t|t-1; r), R_u(t)),$$

where $R_u(t)$ is the achieved rate for client u during scheduling round t (see Definition (7.1) on page 113).

Remark 7.8: *CDF based algorithm for clients with jitter requirements*

In order to incorporate also jitter requirements into the CDF based algorithm, we have to include a control point after step 6, should the resource bin have been allocated to a jitter sensitive client.

If the allocation *reduces* the success probability[4] for client u, then revoke the resource bin, and exclude client u from further allocations during the current scheduling round. This check is only necessary for clients with jitter requirements.

■

7.4.2 The Score Based Scheduling Algorithm (SBA)

This algorithm is inspired by the one proposed by Bonald in [18]. Our algorithm differs from Bonald's in that we use also the transmission rate statistics in the criterion, through the failure probability P_u^F, in order to incorporate delay and throughput requirements, whereas Bonald only uses the channel statistics in order to achieve a *fair* allocation.

Basically, our score based algorithm is the same as the CDF based scheduling algorithm from Section 7.4.1, differing only in how we obtain the values

[4]That is, if the failure probability is larger when calculated for the required rate *after* the allocation, than *before* the allocation.

$J_{\mathcal{C}}(u)$ and $J_{\mathcal{R}}(u)$. Given that we have historical values of the channel capacities $C(u,s)$ and the transmission rates $R_u(t)$, we maintain a *sorted* array of each for each client u,

$$\mathbf{C}_u = \begin{bmatrix} C_{u1}, & ..., & C_{uK} \\ t_1, & ..., & t_K \end{bmatrix}, \tag{7.16}$$

and

$$\mathbf{R}_u = \begin{bmatrix} R_{u1}, & ..., & R_{uL} \\ t_1, & ..., & t_L \end{bmatrix}. \tag{7.17}$$

In (7.16), K is the number of stored past bin capacities. Note that *all* bin capacities $C(u,s)$ from a sliding window of past bin capacities should be stored in (7.16). Thus, $K = NS$, where S is the number of resource bins for each t_i, and N is the number of different t_i. The elements C_{ui} are sorted in *increasing order of bin capacity.* Alongside with the bin capacities C_{ui}, we have time indices t_i, that indicate the age[5] of the bin capacities. They are required in order to update $\mathbf{C}_u$, by *removing the oldest* observations, and *inserting the most recent* ones into position according to their channel capacity value. The size $K = NS$ of the array, is chosen to be large enough to maintain statistics of the small-scale fading variations, which should comprise a duration of 5-10 fading dips. Furthermore, the bin index s has been dropped.

The array (7.17) is defined in a similar way, with the past achieved transmission rates from Definition 7.1 in R_{ui}, sorted in *increasing order of achieved transmission rate* and their corresponding age in t_i. $\mathbf{R}_u$ is updated in the same fashion as $\mathbf{C}_u$.

Having $\mathbf{C}_u$ and $\mathbf{R}_u$, we obtain $J_{\mathcal{C}}(u)$ and $J_{\mathcal{R}}(u)$ in (7.15) as follows:

1. Given each clients' best resource bin, $s_u \in \mathcal{S}_n$, and their respective bin capacities $C(u, s_u)$, find the position i, $\quad 0 \leq i \leq K+1$ in which it would enter the sorted array $\mathbf{C}_u$ in (7.16).

2. For each client u, calculate the probability P_u^C of $C(u, s_u)$ being a good resource
$$P_u^C = \frac{i}{K+1}$$
and set $J_{\mathcal{C}}(u) = P_u^C$.

[5]The time t_i represents either a timestamp of the insertion time into the array, or a counter that is incremented for each scheduling cycle.

3. Given the required transmission rate $\tilde{R}_u$ (7.2) for each client u, find the position $j, \quad 0 \le j \le L+1$ in which it would enter the sorted array $\mathbf{R}_u$ in (7.17).

4. Calculate the failure probability P_u^F for each client u as

$$P_u^F = \frac{j}{L+1}$$

and set $J_{\mathcal{R}}(u) = P_u^F$.

5. Use $J_{\mathcal{C}}(u)$ and $J_{\mathcal{R}}(u)$ in the multiplicative criterion (7.15), make the decision $d(s_{\hat{u}}) = \hat{u}$, update the buffer level $B_{\hat{u}}$, update the set of admissible resource bins $\mathcal{S}_{n+1} = \mathcal{S}_n \setminus s_{\hat{u}}$, and repeat until all resource bins have been allocated.

In Table 7.2, we estimate the complexity for the SB algorithm. Comparing

Operation	Count	Iterations	Complexity	Total
$\tilde{R}_u = B_u/T_u$	U	1	1	U
$C_u = \max_s C(u,s)$	U	1	S	$U \cdot S$
$C_{\hat{u}} = \max_s C_{\hat{u}s}$	μ	S	$S/2$	$\frac{\mu S^2}{2}$
$J_u = J_{\mathcal{R}}(u) \cdot J_{\mathcal{C}}(u)$	U	S	1	$U \cdot S$
$\max_u J_u$	1	S	U	$U \cdot S$
$B_{\hat{u}} = B_{\hat{u}} - C_{\hat{u}s}$	1	S	1	S
$\tilde{R}_{\hat{u}} = B_{\hat{u}}/T_{\hat{u}}$	1	S	1	S
Total				$U + 3US + \frac{\mu S^2}{2} + 2S$

Table 7.2: Number of operations for the SB algorithm. According to this calculation, the score based algorithm is equally complex as the ITER algorithm. Also here, $1 \le \mu \le U$ equals the number of clients $\tilde{u}$, that also had bin $s_{\hat{u}}$ as their best candidate resource bin.

the complexity of the SBA algorithm with that of the ITER algorithm (Table 6.2 on page 101), we find that they both require the same number of operations. What we saved by not involving a weighting factor P_u ($S+U$ operations), we lost by having to calculate and update the required rate $\tilde{R}_u$. Note that we have omitted the updating of the statistics in (7.16) and (7.17) from the complexity analysis, since this may be implemented outside of the scheduler.

Remark 7.9: *Score based algorithm with jitter*

In order to incorporate also jitter requirements into the score based algorithm, introduce a third index, k, that corresponds to the maximum required rate, calculated accordingly in step 3, and let j correspond to the minimum required rate. Then calculate the jitter success probability $P_u^{S_j}$ as

$$P_u^{S_j} = \frac{k-j}{L+1},$$

set $J_{\mathcal{R}}(u) = 1 - P_u^{S_j}$, and use this in (7.15).

If the allocation *reduces* the success probability[6] for client u, then revoke the resource bin, and exclude client u from further allocations during the current scheduling round. This check is only necessary for clients with jitter requirements.

■

7.5 Summary

In this chapter, we have proposed a scheduling criterion, based on the probability of succeeding to meet the service requirements assigned by the service level controller. In order to design (sub-) optimal on-line scheduling algorithms, we have chosen to proceed with a simplification of the otherwise combinatorial optimization problem. We considered combining the individual failure probability (7.4) with the probability of a resource bin being good (7.12). Based on this simplification, we outlined two algorithms, the CBS algorithm, and the SBA algorithm.

The SBA algorithm is equally complex as the ITER algorithm, which makes it an equally attractive candidate in a possible implementation. It remains to evaluate its performance, which will be done in the following two chapters. In Chapter 8, we will briefly look at its performance in a rather simple scenario, whereas more realistic simulations will put the SBA scheduling algorithm to the test in Chapter 9.

[6]That is, if the value $k - j$ is smaller when calculated for the required rate *after* the allocation, than *before* the allocation.

Chapter 8

Simulations

This chapter describes various simulation experiments that have been conducted on the different scheduling approaches proposed in previous chapters, as a preparation for the case study in the next chapter. First, some simplifying assumptions are made, in order to make the analysis tractable. Secondly, two channel models are introduced; a non-correlated Rayleigh fading channel, and a more realistic channel model emulated from ray-tracing simuations of a radio propagation environment. The emulated channel models will be used for the case study in the next chapter. Furthermore, a set of measured real-life radio channels are introduced. Thirdly, we compare the proposed scheduling methods using the non-correlated channel model and the measured channels, in order to also observe how the channel correlation affects the scheduling results.

We hope to demonstrate the benefit of introducing a wireless network with flexible resources, such as the OFDM system described in Section 4.3.5, and show how scheduling and SLC are supposed to cooperate, in the presence of pricing models, in order to maximize revenue for a network operator.

In order to evaluate whether the scheduler and admission control work properly together, we introduce a simple admission control, in the form of a PID service level controller, that will adapt the requirements according to the variations of the measured channels. We will also vary the PID parameters in order to illustrate their effects on the service level control.

8.1 Assumptions

For the purpose of simplifying the analysis of the suggested and exisiting scheduling algorithms some assumptions are made, that limit the implications from secondary issues that are not important for the scheduling itself.

8.1.1 Data Traffic

Leaky-bucket style traffic shaping is assumed to be performed at the wireless network before reaching the wireless link scheduler. This approach will punish *non-conformant* (see Definition 1.1 on page 4) traffic by dropping packets, thereby preserving resources for conformant traffic, when network usage is high. [1]

8.1.2 Wireless Transmission

In the analysis we have not explicitly taken into account the possibility for failure in the reception of the transmitted data. Thus, we do not evaluate the delay effects of retransmissions nor the effects of lost data. However, when data is lost or received in error, a practical system would react by retransmitting the missing data as soon as possible after the transmitter learns about the failure. This may be modeled as an extra traffic load that depends on the prediction error and the target error rate. The simplest way to take this into account is to assume that some percentage of the admitted traffic is used for retransmissions.

8.1.3 Control Signalling

Control signalling is a vital component for the provision of channel-adaptive scheduled transmissions. Without it, there would not be enough knowledge to make an efficient scheduling decision, nor would it be possible to communicate the schedule to the involved terminals. In the following analysis we assume that the control signalling required can be provided accurately.

---

[1] Whether this is part of the admission control or not, is a separate question. One admission setup could reject new sessions at the edge router, when entering the wireless network operator's backbone, thereby relieving it from the work of transporting traffic that would be rejected later due to high wireless link load. Another setup could buffer new sessions temporarily at the wireless access point, cherishing a hope for soon-to-come released resources. In any case, the scheduler is assumed to be presented with traffic that it expectedly can handle.

8.2 Channel Models

We have chosen to run the simulations on three different kinds of channel models:

- One is a block Rayleigh fading channel model with given mean values and variances, with the fading being uncorrelated in both time and frequency,
- the second model is created from real world channel sounding measurements, in urban and suburban environments, and
- the third type are emulated channels, obtained by means of ray tracing calculations, that provide channels with similar properties as the sounding measurements, but with longer duration.

The resulting channel Signal to Noise[2] Ratio (SNR) samples will be used for scheduling purposes, either directly by scheduling with respect to the channel SNR, or by scheduling with respect to the feasible transmission rate that the SNR allows for a given target Bit Error Rate (BER).

All three channel models are based on one general model that we now describe.

8.2.1 General Channel Model

We will use a general model for the time- and frequency-varying SNR, in a dB scale. It consists of two components (see Equation (8.1)):

- The small-scale fading component, $\Gamma(t, f)$, is the time- and frequency-varying component. This component has mean value equal to zero decibel ($0dB$), where the mean value is calculated over the whole simulation.
- The path loss component, $\bar{\Gamma}(l(u))$, is an average SNR value that is imposed on the channel model for client u. It will depend on the distance between the mobile and the base station.

Combining the two components we obtain

$$\Gamma(u, t, f) = \bar{\Gamma}(l(u)) + \Gamma(t, f). \tag{8.1}$$

[2]We assume that the noise is white and Gaussian.

We set the average SNR, $\bar{\Gamma}(u)$, according to where in the cell we wish to place the mobile, and then impose the fast varying characteristics to reflect different travelling speeds and mobile environments.

Remark 8.1: *Explicit interference and shadow fading omitted in (8.1)*
We have not included any explicit interference or shadow fading model in (8.1). We assume all such variations are included in the small-scale fading component $\Gamma(t, f)$.

■

User Distribution and Path Loss

We wish to model a number of users, with positions that are selected according to a pattern that corresponds to a uniform distribution, over the cell area. However, the average SNR values are restricted to be within certain limits, in order to avoid extreme cases. We have chosen to set the maximum and minimum average SNR value due to path loss to reflect a distance relation of 10 times for the nearest and farthest users. The users will thus have average SNR values that depend on their relative distance to the base station. We utilize the path loss model

$$\bar{\Gamma}(l(u)) = \bar{\Gamma_0} + 10log_{10}(l(u)^{-\alpha}), \tag{8.2}$$

where the received power decreases with $l^{-\alpha}$, l being the distance between transmitter and receiver and α is the path loss exponent, here chosen to be $\alpha = 3.5$ which is reasonable for an open terrain and a high antenna [31]. The user distribution model used is

$$N \propto l^2, \tag{8.3}$$

where the number of users N within a distance interval $[l, l + \Delta l]$, increases as the square of the distance l. The path loss model with $\alpha = 3.5$ will imply a difference in average SNR between the nearest and farthest users of 35 dB. We will use an average SNR of 40 dB for the nearest user, and 5 dB for the farthest. The user distribution model will be used by setting the distances of the different users to be

$$l(u) = 1 + (10 - 1)\sqrt{\frac{u-1}{U-1}}, \qquad u = 1, ..., U \tag{8.4}$$

where u is the user index, and U is the number of users in the simulation. With this setting of l, using $\bar{\Gamma}_0$ equal to 40 dB, will yield the desired average

SNR properties for the U different users. We illustrate the usage of the user distribution and path loss model in Example 8.1.

Example 8.1: Average SNR distribution for five users

---

1. Set $U = 5$
2. For $u = 1, ..., U$ calculate $l(u)$ according to (8.4)
3. For each $l(u)$ calculate $\bar{\Gamma}(l(u))$ according to (8.2), using $\bar{\Gamma}_0 = 40$ and $\alpha = 3.5$

u	1	2	3	4	5
$l(u)$	1	5.5	7.36	8.79	10.0
$\bar{\Gamma}(l(u))$ [dB]	40	14.09	9.65	6.95	5

Table 8.1: Distances according to (8.4) and corresponding average SNRs according to (8.2), for five users.

Remark 8.2: *Deterministic user distribution*

The purpose for having a deterministic distribution of users, is to be able to see detailed effects of the different distances, on the outcome of the simulated cases. Having a random user distribution, we would have to run many simulations, in order to find cases as “bad” as our deterministically selected case.

■

8.2.2 Non-correlated Rayleigh Channels

In this model, the SNR values are assumed constant within each time-frequency bin, and uncorrelated between neighbouring bins, in both time and frequency. The instantaneous SNR values are given by (8.1), using a random number x for each bin

$$\Gamma(t, f) = \sigma x$$

yielding

$$\Gamma = \bar{\Gamma} + \sigma x. \tag{8.5}$$

Here, $\bar{\Gamma}$ is the average SNR, according to (8.2), and x is calculated from the uniformly and independently distributed random number n, through the inverse of the Rayleigh cumulative distribution function:

$$x = 10 \log \left(\sqrt{-2 \log (1 - n)} \right) \tag{8.6}$$

The variance σ^2 in (8.5) has been selected to be 1, since this is in accordance with what the measured channels indicate.

8.2.3 Real-world Fading Channels

These channel data sets have been obtained by means of channel sounding measurements. The measurements were performed in the Stockholm area by Ericsson Radio Systems AB. Each measured sequence contains 1430 observations of a 120-tap complex impulse response, at 1880 MHz, with vehicle velocities varying from 30 to 90 km/h. The 120 taps are estimated from 700 channel samples, sampled at 6.4 MHz leading to a channel impulse response rate of 9.1 kHz (a new impulse response is obtained every 0.11 milliseconds). More detailed information about the measurements can be found in Chapter 3 of [29].

For the purpose of generating samples of a wideband OFDM-channel, each 120-tap impulse response was transformed into the frequency domain at 640 points (subcarriers with 10 kHz spacing), out of which the centrally located 500 subcarriers are used in the studied system. Each frequency domain sample in this data set corresponds to one symbol in the OFDM system, giving rise to fading data for $500 \times 1430 = 715,000$ symbols. If we would map this data directly to the example OFDM system outlined in Section 4.3.5, we would have to divide it into time-frequency bins of 120 symbols each, resulting in data for 5958 ($= 25 \times 238$) time-frequency bins, which corresponds to a simulated time of 0.16 s [3].

In the preliminary (preparatory) simulations presented in Section 8.3.2, we have used this data with a modification: Instead of regarding each sample in the time domain as a symbol sample, we have regarded it as a time-frequency bin sample, thereby extending the simulated time to 0.95 s. This

[3] Each time-frequency bin consists of 6 symbols in the time domain and 20 subcarriers in the frequency domain. The symbol period of $111\mu s$, including cyclic prefix, and the carrier spacing of 10 kHz result in time-frequency bins of 0.667 ms length and 200 kHz width.

is not good modus operandum in the general case, but the purpose was only to see how correlation in time and frequency would affect the scheduler. The implication of doing this is simply that the channels behave as if all the mobile receivers had been traveling at 1/6 of their real speeds.

8.2.4 Emulated Channels

For a more detailed study of how the system proposed in Chapter 9 would behave, we have emulated a number of mobile channels. The channel emulation has been performed by means of ray tracing complex sinusoids originating from a virtual antenna placed at the origin of a coordinate system. The sinusoids are ray traced over multiple paths, scattered by virtual objects at different locations in the coordinate system, to finally be combined in a moving virtual receiver. The channel emulator is a result of a Master's Thesis project [13], and it has been verified against real channel sounding data from [29]. It includes modelling of path loss, reflection loss (absorption), reflection phase shift, optional LOS (Line Of Sight) component, multipath propagation, and also shadow fading (although modeled as an autoregressive process, and not as a result of obstructing virtual objects).

In Figure 8.1 we see the result of emulating 13 channels, and in Table 8.2 the used parameters are presented. The *Position*, *Speed*, and *Direction* parameters, together define how the mobile receiver will perceive the received power from the base station that resides at the origin of the coordinate system. There are also other attributes to the emulation of the channels that we do not present in great detail, such as the scattering environment and the autoregressive model for the shadow fading, but they can be provided upon request.

In Figure 8.1 we present the obtained channel SNRs. Since the emulated channels come from mobiles that are at different distances from the base station, they will result in different average SNR values. However, since we wish to utilize the same mobile properties in different simulations, where the mobiles can be placed arbitrarily in the cell, we would like to remove this average SNR from the emulated channels, and instead utilize an imposed average SNR. Therefore, the emulated channels presented in Figure 8.1 each have an average SNR of 0 dB. The different average SNRs that will be used later will be imposed afterwards, by adding a desired average value, to reflect the different distances from the base station. This average SNR will be added in a similar fashion as $\bar{\Gamma}$ for the non-correlated channels, according to Equation (8.5). The difference to the non-correlated case is that instead of using σx from a log-Rayleigh distribution, we now calculate the received

Channel	Position	Speed	Direction	Doppler
1	(0,800)	90 km/h	290^o	157 Hz
2	(250,400)	90 km/h	100^o	157 Hz
3	(300,750)	70 km/h	225^o	122 Hz
4	(57,555)	11 km/h	350^o	19 Hz
5	(0,450)	90 km/h	45^o	157 Hz
6	(194,658)	1 km/h	210^o	1.7 Hz
7	(500,600)	100 km/h	176^o	174 Hz
8	(300,900)	120 km/h	240^o	209 Hz
9	(0,200)	130 km/h	85^o	226 Hz
10	(200,450)	30 km/h	120^o	52 Hz
11	(-130,695)	30 km/h	175^o	52 Hz
12	(0,100)	60 km/h	45^o	105 Hz
13	(40,676)	5 km/h	0^o	8.7 Hz

Table 8.2: In order to create channel data using ray tracing models, we need to provide exact information of the mobile terminals' positions and velocities, the environment and its electromagnetic properties, the transmission frequency and bandwidth, and the properties of the base station antenna. This table presents the input data that is characterizing for each of the 13 emulated channels. *Position* is the initial position in meters in a coordinate system where the base station resides in the origin. The mobile then travels at the velocity *Speed* in the direction given in *Direction*, resulting in a maximum Doppler frequency according to *Doppler*.

power from the ray tracing model of the mobile environment.

The emulator created channels of 6.4 MHz bandwidth over 32 frequency bins, out of which the center 25 have been selected for the purpose of creating the 25 frequency bins over the 5 MHz wide carrier band. The selection of a 5 MHz wide carrier band was due to that the proposed OFDM system, as described in Section 4.3.5, uses this bandwidth to agree with current 3G frequency allocations. The emulated channels are sampled at a rate of 0.667 ms, which is the time slot length in the proposed OFDM system. The length of the sequences is 22492 samples, giving a duration of 15 sec.

8.3 Scheduling Based on the Quadratic Criterion

In this section we will investigate the properties obtained when using the approximations (6.12) of the quadratic criteron, embodied in the MAXR and ITER scheduling algorithms. We also investigate the search based CSD

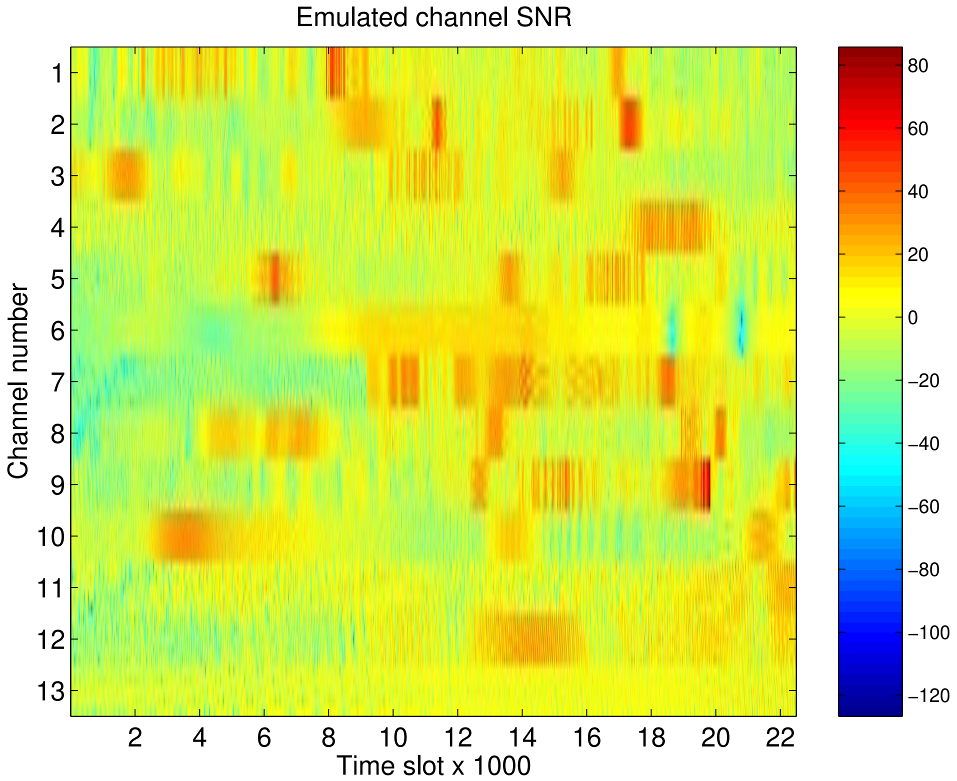


Figure 8.1: The 13 emulated channels are here represented as 13 horizontal bands with their channel numbers along the vertical axis. Within each of the channels there are 25 frequency bins, also distributed along the vertical axis, each having its own SNR value. Each frequency bin consists of 20 subcarriers with 10 kHz spacing, assumed to have equal SNR values within the bin. Along the horizontal axis we have the time slot numbers. One time slot corresponds to 0.667 ms, resulting in 22492 time slots for a 15 seconds long channel trace. Each time slot has its own SNR value, and consists of 6 symbols per subcarrier. This results in a time-frequency bin consisting of 20 subcarriers with 6 symbols each, giving a total of 120 symbols per time-frequency bin. Thus, each sample corresponds to a channel SNR value, valid for 120 symbols, where the different colors represent the channel SNR values according to the scale at the right. All the emulated channels have here been normalized to have a 0 dB average SNR. The image has lower resolution in both time and frequency than the data itself.

scheduling algorithm, that uses the original quadratic criterion (6.2). The al-

User	1	2	3	4	5
SNR	40.0	14.1	9.65	6.95	5.00

Table 8.3: The average SNR values for users at different distances from the base station. They have been chosen such that the nearest user experiences an average SNR of 40 dB and the fartherst user experiences an average SNR of 5 dB. Between these boundary values, the SNRs are selected according to Equations (8.4) and (8.2).

gorithms were presented in Section 6.3.2 to Section 6.3.4 respectively. Here, we will compare the performance of the algorithms using both correlated and non-correlated channel models, in order to study the impact on the scheduling performance, of the correlations, of increasing traffic loads, and also of including a simple admission control.

8.3.1 Non-correlated Rayleigh Fading Channel Model

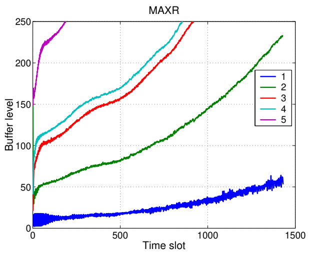
We present the average buffer level trajectories over 100 simulations with the same set of parameters for each scheduling algorithm. The input data rate to the system (the admitted traffic) is increased linearly from 1.54 Mb/s[4] up to 1.85 Mb/s per user, from time slot 1 to time slot 1430. There are five active users in the system, placed at different distances from the base station, thereby experiencing different average SNR. The average SNRs have been imposed according to (8.5), with the average values according to Table 8.3, which are the same as in Example 8.1. Furthermore, as the title indicates, the small-scale fading channel models used in this section, are the non-correlated channels described in Section 8.2.2.

No weighting factor

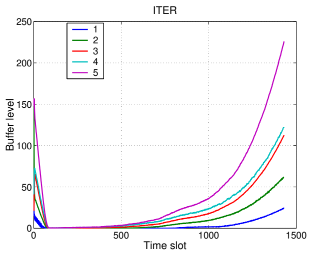
The first set of simulations illustrates the importance of a working admission control, that can balance the load appropriately. Furthermore, we will see evidence of the expected problem with the MAXR algorithm, as mentioned in Section 6.3.6 and Figure 6.5 on page 107, due to its crude approximation of the quadratic criterion.

Studying Figures 8.2(a) through 8.2(c), we see how the scheduling algorithms MAXR, ITER, and CSD, respectively, perform on the buffer levels. They are implemented according to the descriptions in Section 6.3.2 to Section 6.3.4. A number of interesting observations can be made from these

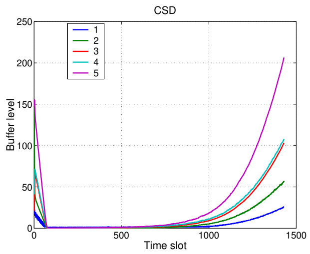
[4] 1 Mb/s = 2^{20} bits/second



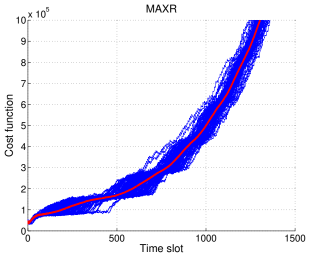
(a) Buffer levels, MAXR



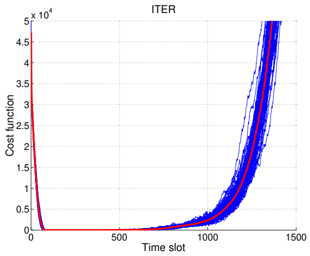
(b) Buffer levels, ITER



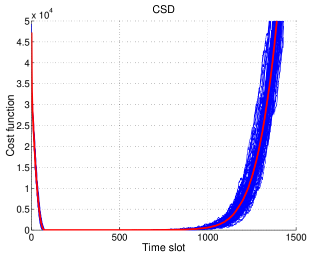
(c) Buffer levels, CSD



(d) Cost function, MAXR



(e) Cost function, ITER



(f) Cost function, CSD

Figure 8.2: Buffer levels and cost functions for five clients. We compare three scheduling algorithms, when increasing the load from 1.54 Mb/s to 1.85 Mb/s for each client. In the top figures, the SNR (or distance in this case) for user 1 to 5, is defined in Table 8.1.

figures:

1. CSD keeps the buffer levels lower than the other two algorithms, although ITER is not much worse.

2. The buffer levels are indirect measures of the queueing delays. Therefore CSD can maintain lower delays than the other two algorithms.

3. The buffer levels are different for the different users. This is due to the worse average channel for the fartherst user, which results in a lower criterion value for a given buffer level.

4. CSD manages to keep the buffer levels more homogeneous than the other two algorithms, thereby resulting in a more equal service for the different users.

5. MAXR fails to maintain low buffer levels, probably due to its crude approximation of the quadratic criterion in the case when few users compete for many resources.

6. Common to all scheduling algorithms is the problem of not being able to meet the increasing resource demand as the input rates to the buffers increase, as the time index increases.

From observation 1 and 2 we conclude that we would prefer the CSD algorithm in terms of performance. The second choice would be the ITER algorithm. From observation 3 we deduce that there is a need for the weighting factor P_{ui} discussed in Section 6.1.1, if we wish to give the different streams using the same service equal delays, which we do, since it would otherwise not be the same service. Observation 6 indicates that in order to provide predictable services, we need to combine scheduling with admission control.

Furthermore, from the resulting cost function values in Figures 8.2(d) through 8.2(f) we draw the following conclusions:

1. CSD keeps the cost function lower, for higher loads, than the other algorithms.

2. ITER accomplishes an almost equally good result. At least it gives a significant improvement over MAXR in terms of buffer level and cost function value.

3. The cost function trajectories deviate little from their mean. This is true for all three algorithms, but particularly clear for the CSD algorithm.

4. The problem with the MAXR algorithm has to be solved if we wish to use it for few users. Otherwise MAXR will only be useful for scheduling many users on few resources.

Channel quality weighting factor

In the following simulations, we will incorporate a weighting factor P_{ui} with respect to the experienced channel quality. In this set of simulations, we calculate the weighting factor on-line, based on the previous transmissions, according to

$$\frac{1}{P_{ui}(T)} = \frac{\sum_{t=T-40}^{T} a_u(t)}{\sum_{t=T-40}^{T} \sum_{s=1}^{S} \delta(d_s(t) - u)} \qquad , u = 1, ..., U, \tag{8.7}$$

where $a_u(t)$ is the allocation vector entry for client u in scheduling round t, $d_s(t)$ is the decision vector entry s, also in scheduling round t (see the definitions in Equations (4.5) and (4.6) on page 61). Here, $\delta(x)$ is the delta function, defined in Equation (4.3) on page 59. Thus, in (8.7) the numerator

$$\sum_{t=T-40}^{T} a_u(t)$$

represents the total alloctaion during the last 40 scheduling rounds to user u, in terms of BPSK bins[5], whereas the denominator

$$\sum_{t=T-40}^{T} \sum_{s=1}^{S} \delta(d_s(t) - u)$$

is the total number of resource bins allocated to client u, during the previous 40 scheduling rounds of the simulation. The division in (8.7) then results in an estimate of the current average channel throughput each client experiences when accessing the channel.

This weighting factor will have the effect of weighting up the cost of the farther users relative to the nearer users, so that the farthest user will get access to the channel before its buffer level grows much higher than the nearest user's. What is actually done, is that each user's estimated bin capacity in each time-frequency bin, $C(u, s)$, is divided by their sliding window average allocated bin capacity, C_u, so that they get values larger than 1 for *relatively* good time-frequency bins, and values smaller than 1 for relatively bad bins.

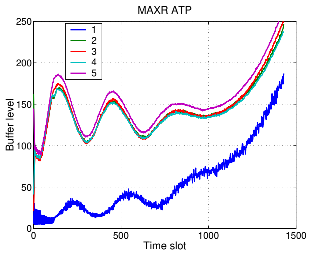
Introducing the weighting factor P_{ui} that compensates for each stream's average experienced throughput, we obtain Figures 8.3(a) through 8.3(f), that show the same quantities as the Figures 8.2(a) through 8.2(f), but now using P_{ui}.

From Figures 8.3(a) to 8.3(f) we make the following observations:

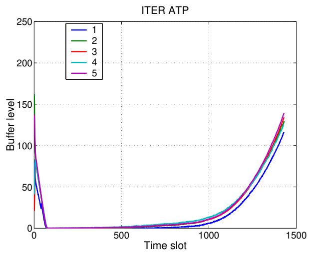
1. We have reduced the queueing delays for the fartherst users, at the expense of increasing it for the nearest users. This was the intention for introducing the weighting factor.

2. The introduction of the weighting factor did result in a worse average performance of the schedulers. The average remaining buffer levels at the end of the simulations increased slightly for an average stream using the ITER and CSD algorithms, meaning that the average delay

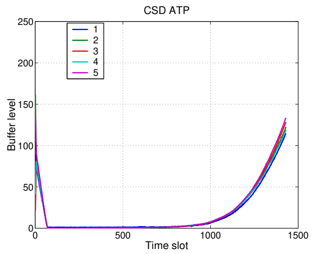
[5]Recall from Section 4.2 that $C(u, s) = 1$ means uncoded BPSK modulation.



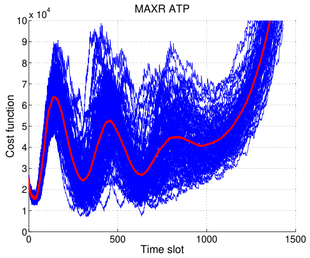
(a) Buffer levels, P_{ui}-weighted MAXR



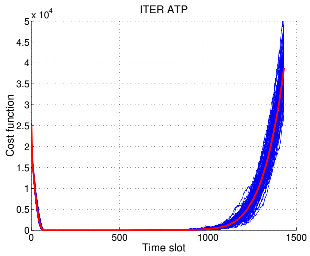
(b) Buffer levels, P_{ui}-weighted ITER



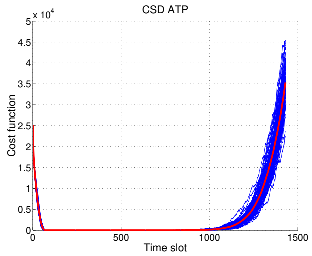
(c) Buffer levels, P_{ui}-weighted CSD



(d) Cost function, P_{ui}-weighted MAXR



(e) Cost function, P_{ui}-weighted ITER



(f) Cost function, P_{ui}-weighted CSD

Figure 8.3: Buffer levels and cost functions for five clients. We compare the *average channel quality* compensated quadratic criterion algorithms, when increasing the load from 1.54 Mb/s to 1.85 Mb/s for each client.

increased slightly. Intuitively, this makes sense, since the cost function without weighting factor acts more directly on the buffer levels, than a weighted version of it.

3. The MAXR algorithm performs significantly better with weighting factor, than without (note the exponent on the cost function). However, MAXR fails in equalizing the buffer levels for all users, despite the weighting factor.

4. Furthermore, MAXR shows an oscillative behaviour that is due to its crude quadratic criterion approximation, that becomes most apparent with a relatively stationary channel. In Observation 8.1 we expound on this problem in some more detail.

Observation 8.1: *MAXR oscillations with stationary and flat fading channels*

When a client has a channel whose bin capacities do not overlap with other channels' bin capacities (such as client number 1 in Figure 8.3(a)), its criterion will grow without the client receiving any resources, until the buffer level reaches a value where it influences the criterion to exceed all the other users' contributions. Since the channel is rather static, it will then have a high criterion value for all resource bins, and will therefore receive all resource bins in that scheduling round. This is a shortcoming of the non-updated gradient in the MAXR algorithm (due to the buffer levels, or token levels, not being updated between each allocation of a resource bin).

■

Apart from the oscillatory buffer levels, another problem may arise, requiring attention for these cases, when using the MAXR algorithm. This will be described next.

Observation 8.2: *Simple re-allocation avoids resource waste and mitigates oscillations*

It is very likely that the MAXR algorithm allocates too many resources to a client, once it decides to allocate to a client with a high and exceptionally constant criterion (flat and slow fading) for all resource bins, when meeting cases as the one observed in Observation 8.1. This may lead to resource bins being unused, and thereby wasted, since the served client does not have the amount of data required to fill the resource bins. The MAXR algorithm should therefore be complemented with a simple re-allocation to avoid resource waste in these cases. The Robin Hood algorithm from [32], also described in Appendix A.2, provides such a simple re-allocation.

■

Conclusion for non-correlated channel simulation

What the previous simulations tell us, is that token bucket levels can be controlled by the service level control algorithm, by means of adjusting the admitted data flows by adjusting the token rates. In this example, we used a faked service level controller which continuously increased the token rate. In a real application, this would of course not be the case, since the service level controller would withdraw, should the scheduler fail to empty the buffers in a timely manner.

As long as the weighting factors are accurate, they will provide a normalization for the scheduler's criterion function, so that the buffer levels automatically maintain a given relation in size. Controlling the actual size will be the task for the service level control algorithm. We will next test this by introducing the measured channel data.

8.3.2 Performance in the Presence of Correlation in Time and Frequency

In the following simulations we will explore how the scheduling algorithms behave when they are applied on more realistic channels. These channels encompass properties such as high correlation in time and frequency, which may be challenging for the schedulers. The time and frequency correlations have the effect of maintaining both good and bad conditions over long (several milliseconds) time spans.

In the simulations that we present next, we will use the channel SNRs presented in Section 8.2.3, of which we see a five-user example in Figure 8.4. The measured channels have been transformed according to the user distribution and path loss model described in Section 8.2.1.

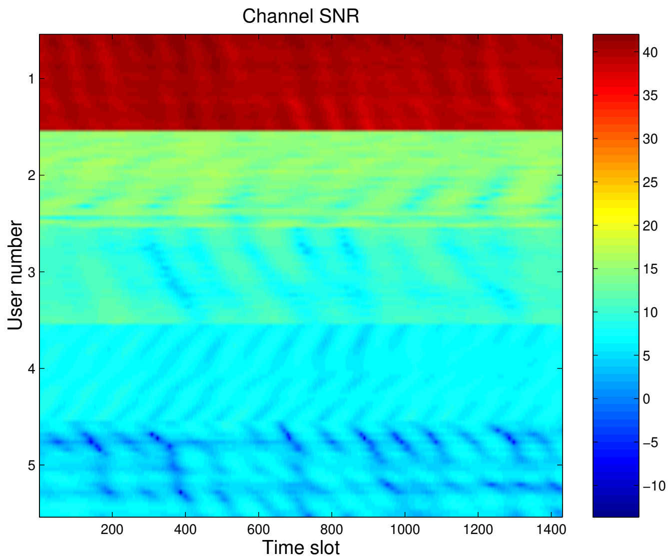


Figure 8.4: Channel SNR levels for the five users.

Channel quality weighting factor

We will first try the same experiments as in the previous section on a set of real-world measured channels. These are quite short in time, but we will be able to see the possibilities, and pitfalls, encountered for longer channels. We have a set of 45 channel measurements, out of which we randomly pick five, and run the simulation. This is repeated 100 times and the result is averaged and presented in Figure 8.5 through Figure 8.10. The input data rates have been set to increase from 1.54 Mb/s up to 1.85 Mb/s for each user, throughout the 1430 time-slots.

In the first three figures, Figure 8.5 through Figure 8.7, we see the average buffer levels over the 100 simulations. What is encouraging, is that the simple MAXR algorithm, on average, seems to be handling the offered load nearly as well as the other two, more complex, algorithms. The results in Figure 8.8 through Figure 8.10 tell us that the cost functions in most of the simulation runs are kept rather low (close to the x-axis), but that some cases are too difficult for the scheduler (the ones that depart from the x-axis). That the scheduler can't entirely handle the load is not only the fault of the scheduler, but also of the admission control, that does not monitor the performance of the scheduler. Instead of adapting the load, the faked admission control increases the load continuously.

Introducing Admission Control

We will now consider letting the service level controller (SLC), that is part of the admission control (see Section 3.3), monitor and control the scheduling performance. This will be done by feeding back to the SLC, the current output buffer levels, and based on that value, adjust the token rate for two of the flows. In the following simulation we have used a simple PID regulator, described in Example 3.6[6] and Figure 3.3. It adjusts the input rate of streams (or users, or sessions, or clients) number 1 and 3, based on their resource cost efficiency and the current load.

The PID controller here takes the maximum output buffer level as a feedback value, and compares it to the target buffer level, thereby obtaining the error signal. This error signal updates the control output, that through client-dependent adaptive gains, K_u (incorporating the cost efficiencies E_u), adjust the admitted token rates for clients 1 and 3, which can be considered "best effort" service clients. In this example the channel-weighted ITER algorithm and the measured channels were used.

[6]The static control gain is here set to $K = 0.004$, and $f(E_u) = \sqrt{E_u}$.

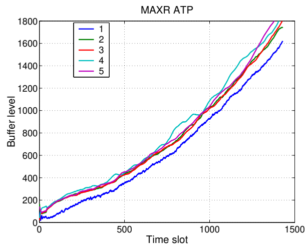


Figure 8.5: Buffer levels for five mobiles, communicating over real measured channels, scheduled with the MAXR algorithm.

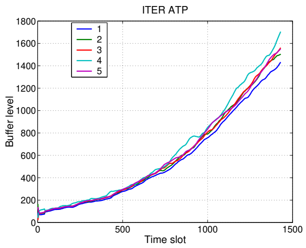


Figure 8.6: Buffer levels for five mobiles, communicating over real measured channels, scheduled with the ITER algorithm.

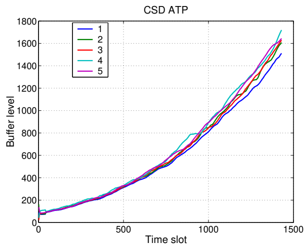


Figure 8.7: Buffer levels for five mobiles, communicating over real measured channels, scheduled with the CSD algorithm.

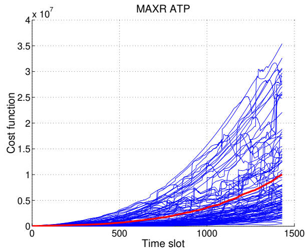


Figure 8.8: Cost function values for 100 simulations, and their average, using the MAXR algorithm.

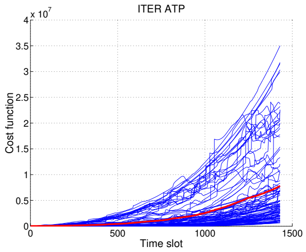


Figure 8.9: Cost function values for 100 simulations, and their average, using the ITER algorithm.

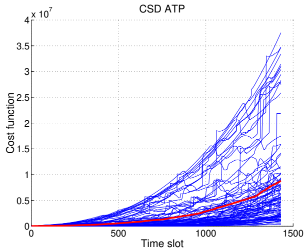


Figure 8.10: Cost function values for 100 simulations, and their average, using the CSD algorithm.

The token rate value is limited both upwards and downwards according to the rate limits in Table 8.4. The target value (the reference value to the PID controller) for the output buffer level is 100 BPSK bins. The resulting buffer levels are presented in Figure 8.11, the token rates controlled by the SLC are presented in Figure 8.12, and the channel SNRs are the ones given for reference in Figure 8.4.

The minimum and maximum rates affect the lower and upper limits of the PID controller, whereas the minimum and maximum prices are addressed by the cost efficiency E_u (see Definition 3.4, on page 49). Note that the cost efficiency E_u also incorporates the rate limits, and that E_u is part of the client dependent gain factor, K_u (see Example 3.6 on page 49), utilized to demultiplex the scalar PID control signal, yielding a single-input, multiple-output, (SIMO) controller.

User	Min rate	Max rate	Min price	Max price
1	0.01	1	0.01	0.4
2	0.2	0.2	0.2	0.2
3	0.01	1	0.01	0.6
4	0.2	0.2	0.2	0.2
5	0.2	0.2	0.2	0.2

Table 8.4: Service level values for the simulation presented in Figures 8.11 and 8.12. Min/max rate refer to the service level breakpoints in the price model for this service class, whereas min/max price refer to the corresponding prices in that same price model. Note that users 2, 4, and 5 have constant values, meaning that they do not accept any flexibility, whereas users 1 and 3 have a span of variability for their service rates and pricing. The price model for users 2, 4, and 5 is thus Model 2, presented in Figure 2.3(b), whereas the price model that best fits users 1 and 3, is Model 4, presented in Figure 2.3(d). This results in an SLC that controls the service rates for users 1 and 3, in order to adapt to the current channel properties.

From Figure 8.11 we see that the SLC manages to maintain the output buffer levels rather constant and that the ITER scheduler also manages to keep them rather equal. We also observe an influence from service level control that needs to be adressed with the weighting factor, by setting the external priority P_{ue} to an adequate value.

Observation 8.3: *Delay variations 1*

If the input rates to the output buffer would be constant, then the queueing delay in the output buffer would also be rather constant and equal. If we regard users 2, 4, and 5 as users with delay constraints, then they can be met in the presented case. However, for user 1 and 3, the delay will vary, since their input rates vary without either adjusting their target buffer levels or adjusting their scheduling weighting factors. This can be seen in Figure 8.13, where the delay for user 3 is much larger than that of user 1, which in turn is higher than that of users 2, 4 and 5.

■

From Observation 8.3 we conclude that if we wish to control also the delay for the users for which we control the rate, then we need some additional mechanism. This can be achieved by setting different external weighting factors P_{ue}, as outlined in Section 6.1.1. Before we do that, we shall however explore the effects of having less resources reserved for fixed rate clients, and allow more of the resources to be adjusted, to meet the channel variations.

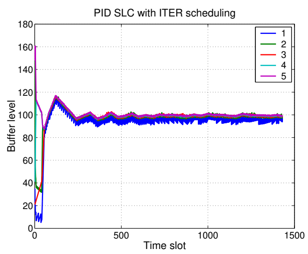


Figure 8.11: Output buffer levels with the weighted ITER algorithm, with a PID controller handling the SLC of users 1 and 3. The target buffer level is 100 for the PID SLC.

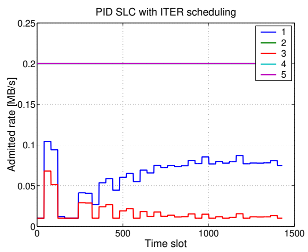


Figure 8.12: Data rates into the output buffer with the PID controlling the service levels (token rates), and the ITER algorithm handling the scheduling.

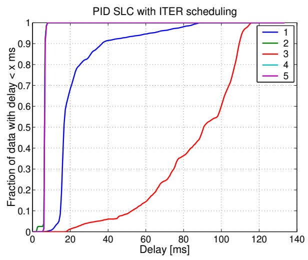


Figure 8.13: Delay distribution for the five users with PID service level control and weighted ITER scheduling algorithm.

Example 8.2: Different service levels

In the following simulation, we have changed the service level parameters for the “fixed rate” users (2, 4, and 5), so that they require only half the rate, leaving more resources for the “best effort” users (1 and 3). The service level parameters are given in Table 8.5, and the channels used are five of the measured channels, described in Section 8.2.3.

We have also changed the scheduling algorithm to the channel-weighted (through P_{ui}, as described in Equation (8.7)) controlled steepest descent (CSD) algorithm, but this does not influence the result significantly.

The resulting buffer levels, input rates, and delay distributions are given in Figures 8.14 to 8.16.

According to the settings in Table 8.5, the SLC will control the service rates for users 1 and 3, in order to adapt to the current channel properties. Compared to the case in Table 8.4 we have here reduced the required rates for the “fixed rate” users.

From Figures 8.14 to 8.16 we note that the increased amount of resources for “best effort” clients allows them to increase their data rates, which in turn reduces their delays:

User	Min rate	Max rate	Min price	Max price
1	0.01	1	0.01	0.4
2	0.1	0.1	0.2	0.2
3	0.01	1	0.01	0.6
4	0.1	0.1	0.2	0.2
5	0.1	0.1	0.2	0.2

Table 8.5: Service level values for the simulation presented in Figures 8.14, 8.15, and 8.16. Min/max rate refer to the service level breakpoints in the price model for this service class, whereas min/max price refer to the corresponding prices in that same price model. We can see that users 2, 4, and 5 have constant values, meaning that they do not accept any flexibility, whereas users 1 and 3 have a span of variability for their service rates and pricing.

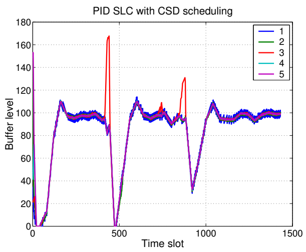


Figure 8.14: Output buffer levels with the weighted CSD algorithm, with a PID controller handling the service level control of users 1 and 3. The target buffer level is 100 for the PID SLC.

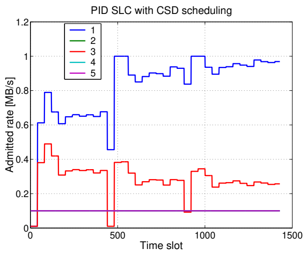


Figure 8.15: Data rates into the output buffer with the PID controlling the service levels (token rates), and the CSD algorithm handling the scheduling.

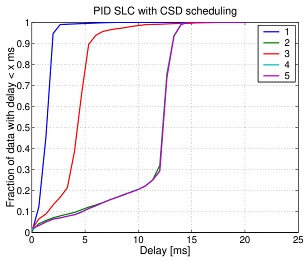


Figure 8.16: Delay distribution for five users with PID service level control and CSD scheduling algorithm.

Observation 8.4: *Delay variations 2*

Since we have reduced the rates of the “fixed rate” users, more resources will be left for the “best effort” users. This is first reflected in the higher rates that they receive by the service level controller (Figure 8.15). Furthermore, since the same buffer level targets were used (100 BPSK bins, see Figure 8.14) as reference value for the PID controller, the higher rates given to users 1 and 3, result in lower queueing delays for them, as seen in Figure 8.16 (note the different resolution in the x-axes for the delay plots in

Figures 8.16 and 8.13, respectively).

■

The high data rates for certain clients result in a high sensitivity in the buffer levels to channel quality variations:

Observation 8.5: *Buffer level variations*

The buffer levels in Figure 8.14 show large deviations from the target around time slots 500 and 1000. This is due to the reduced possible trasmission rate for user 3; its channel enters deep fades at the instances of the deviations. However, should the fades persist, then the service level controller would adapt the token rates to reduce the requirements on the transmission rates.

■

Observation 8.5 also indicates that the external priority P_{ue} should be adjusted according to the admitted rates, in order to avoid such large variations in all buffer levels due to a single user entering a bad channel state.

Introducing the External Priority P_{ue}

We will now introduce an external priority P_{ue}, as also asked for after Observation 8.3, that will make all clients' delays as equal as possible.

Example 8.3: External priority for equal delays

We will use the same service levels as in Table 8.4, the ITER scheduling algorithm, and the measured channels. However, we will introduce the external priority P_{ue}, that we will set according to Equations (6.6) and (6.7) in Example 6.2, on page 91, with the target delay set to 10 time slots $= 10 \cdot 0.667$ ms $= 6.67$ ms , and I_u equal to the admitted rate of each client.

The results are presented in Figures 8.17 to 8.19.

We make a few observations from the figures, concerning delay requirements and buffer levels.

Observation 8.6: *Target delays are well met*

Figure 8.19 clearly shows that the target delay of 6.67 ms is met to a large extent, due to the external priority, that is set to the inverse of the target buffer level, according to Equation (6.7).

■

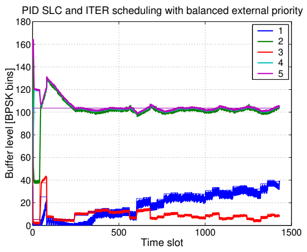


Figure 8.17: Output buffer levels with the weighted and balanced ITER algorithm, with a PID controller handling the service level control of users 1 and 3.

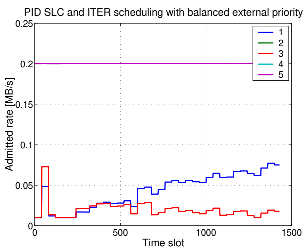


Figure 8.18: Data rates into the output buffer with the PID controlling the service levels (token rates), and the weighted and balanced ITER algorithm handling the scheduling.

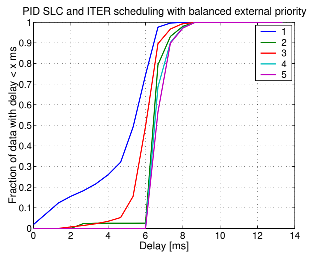


Figure 8.19: Delay distribution for five users with PID service level control and weighted and balanced ITER scheduling algorithm. The target delay is set to 10 time-slots, which equals 6.67 ms.

Observation 8.7: *Buffer levels vary according to admitted rates*

Comparing Figure 8.17 and Figure 8.18 we see that the target buffer levels vary as the admitted rates vary. The target buffer levels are assigned by the admission policy (AP), depending on the delay constraints, and used only by the PID SLC.

■

From these observations we conclude that it is possible to simultaneously control both delay and throughput for certain clients, when having the possibility to adjust the admitted rates of some "best effort" clients.

8.3.3 Flat and Static Channels

In order to qualitatively see the impact of exploiting the variability imposed by the fading, we have run two simulations using $\Gamma(t, f) = 0$ dB in (8.1), thereby yielding completely static channels with fixed SNR:s that depend on the distance to the base station. In the cases presented in Figure 8.20 and Figure 8.21, we have used the ITER algorithm, without and with the channel quality weighting factor P_{ui} from (8.7), respectively. We can make two main observations. The first one motivates the exploitation of the channel variabilities, instead of trying to "average out" the variabilities by means of time and frequency spreading.

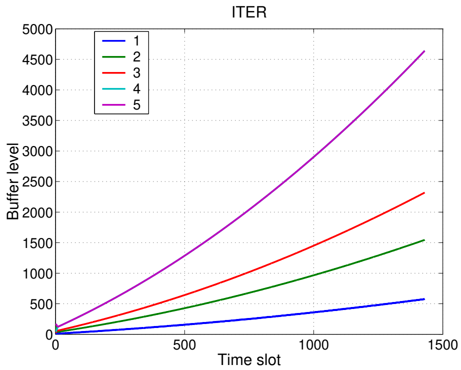


Figure 8.20: Buffer levels for five mobiles, communicating over flat and static channels, with increasing throughput demads from 1.54 Mb/s up to 1.85 Mb/s, scheduled with the ITER algorithm, without weighting factor. On the x-axis we have the time slot number, and on the y-axis, the buffer levels.

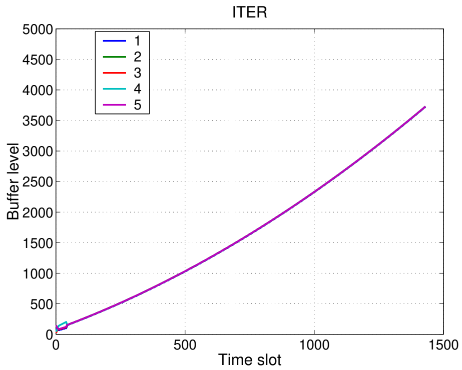


Figure 8.21: Buffer levels for five mobiles, communicating over flat and static channels, with increasing throughput demads from 1.54 Mb/s up to 1.85 Mb/s, scheduled with the ITER algorithm, with on-line updated weighting factor P_{ui}. On the x-axis we have the time slot number, and on the y-axis, the buffer levels.

Observation 8.8: *Flat and static channel gains give worse throughput*

Comparing Figures 8.20 and 8.21 to Figure 8.6, we clearly see that the buffer levels grow much faster in the current flat and static case, than in the correlated fading channel cases. From this we understand that the fading variability is a valuable source for increased link efficiency that should be exploited.

■

The second observation concerns Figure 8.21 and the channel quality weighting factor P_{ui}.

Observation 8.9: P_{ui} *equalizes the buffer levels*

As we can see from Figure 8.21, all five users have approximately equal buffer levels throughout the simulation. This is due to the channel quality weighting factor that in the flat and static channel case manages to equalize the buffer levels almost perfectly.

■

8.3.4 Service Level Control as a Linear Program

Before we investigate the performance of the probability based scheduling algorithm in Section 8.4, we give an example of how the linear program service level controller may perform. We have here used the measured channels and the service level parameters from Table 8.4. See Figures 8.22 to 8.24 for the resulting performance.

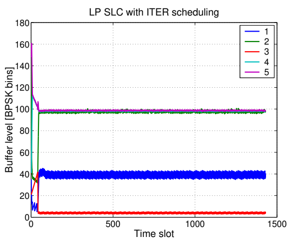


Figure 8.22: Output buffer levels with the ITER algorithm, with a linear program handling the service level control of users 1 and 3. The target delay is 10 time slots, that is 6.67 ms.

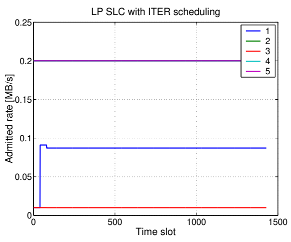


Figure 8.23: Data rates into the output buffer with the linear program the service levels (token rates), and the ITER algorithm handling the scheduling.

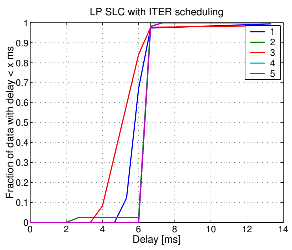


Figure 8.24: Delay distribution for five users with linear program service level control and ITER scheduling algorithm.

Observation 8.10: *Steady service rates*

The LP solution immediately finds the appropriate data rate for the present channel conditions. Since the average bin capacity does not change much during this short simulation, the admitted rates remain optimal throughout the whole simulation.

■

Observation 8.10 promises good performance from the linear programming based service level control. However, we must bear in mind Observation 3.1, from page 53, when the channel properties become more varying during the longer simulations in the case study, in Chapter 9.

8.4 Probability Based Scheduling Algorithm

We will finally present the performance of our score based scheduling algorithm (SBA), presented in Section 7.4.2, that combines probabilistic measures of service urgence and resource qualities.

8.4.1 Performance in the Presence of Correlation in Time and Frequency

We will here repeat the experiment from Section 8.3.2, but now using the SBA algorithm. Also here, we run the simulation 100 times, randomly selecting five out of the 45 available measured channels for each run. The result of this simulation is presented in Figure 8.25, and discussed in Observation 8.11 and 8.12. The first observation concerns the non-stability of the

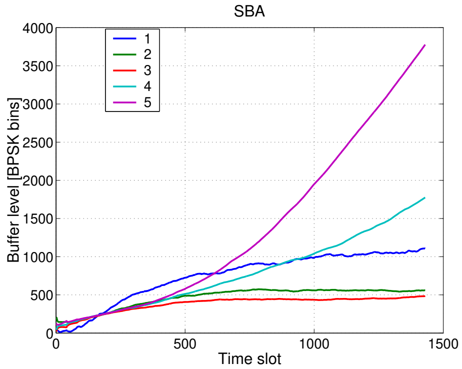


Figure 8.25: Output buffer levels with the Score Based (SBA) algorithm, with the input rate increasing linearly from 1.54 Mb/s to 1.85 Mb/s, from time-slot 1 to 1430, for all users. The simulation has been repeated 100 times with 5 measured channels randomly selected from 45 available measurements. The target delay (see Equation (7.2) in Definition 7.3, on page 114) was set to 20 time-slots (= $20 \cdot 0.667$ ms = 13.3 ms), and the size of the arrays storing historical bin capacities and transmission rates (see Equations (7.16) and (7.17), respectively, in Section 7.4.2) were $K = 25 \cdot L$ and $L = 40$, respectively.

SBA algorithm.

Observation 8.11: *SBA does not keep queue sizes together*

It is clear from Figure 8.25 that the SBA algorithm "gives up" on client number 5 at about time-slot number 500. This is an effect of the limited buffer level function, that is, the failure probability P_u^F. It cannot become larger than 1, and since it is probable that all the other clients also have failure probabilities quite close to 1 at these data rates, SBA will instead discriminate with respect to the probability of a good reource, P_u^C. Thus, the SBA algorithm then reduces to the SB algorithm from [18], described in Section 5.3.2.

■

The second observation tells us that the non-stability might be a desirable feature.

Observation 8.12: *A "released" client is an overload warning*

When SBA "gives up" on client 5, and later also on client 4, it catches up with the other clients, which can be seen in Figure 8.25 as the three lines for clients 1, 2, and 3, level out after time-slot 1000. In order to catch up also with client number 4 and 5, we require an admission control algorithm, that regulates the admitted data rates based on the failure probability, and the cost efficiency, for the different clients.

■

8.4.2 PID Service Level Control

We will now run the SBA scheduler with a PID service level controller (SLC), using the same service level requirements as presented in Table 8.5.

Three things deserve mentioning before presenting the results:

- The SBA algorithm requires explicit delay service requirements, which we have set to 20 time slots ($= 20 \cdot 0.667$ ms $= 13.3$ ms) for all five clients.
- Furthermore, the PID SLC will for the SBA algorithm control the admitted rates according to a target failure probability, $\hat{P}_u^F$, that we have set to 0.5. This means that if the maximum failure probability after a scheduling round differs from $\hat{P}_u^F$, then the admitted rates will be adjusted by the PID controller.
- Since the value that will be fed back to the PID controller here, differs much from the case when the PID regulates with respect to the buffer levels, we have to adjust the scalar control gain K in Example 3.6. By coarse tuning of the PID, we have arrived at the value $K = 0.4$.

The results are presented in Figures 8.26 to 8.28. We can make a number of observations concerning the performance of the SBA algorithm.

Observation 8.13: *Buffer level is not target*

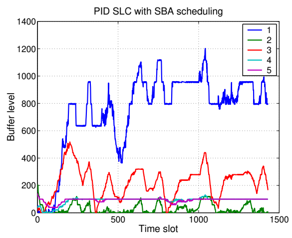


Figure 8.26: Output buffer levels with the Score Based (SBA) algorithm, with a PID controller handling the service level control of users 1 and 3.

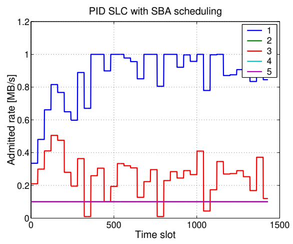


Figure 8.27: Data rates into the output buffer with the PID controlling the service levels (token rates), and the SBA algorithm handling the scheduling.

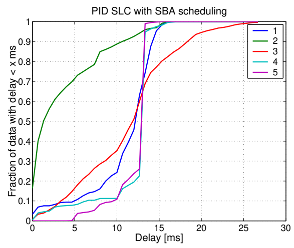


Figure 8.28: Delay distribution for five users with PID service level control and SBA scheduling algorithm. The target delay is 13.3 ms for the SBA scheduler, and the target failure probability ($\hat{P}_u^F$) is 0.5 for the PID SLC.

Comparing Figure 8.26 to Figure 8.14, we see that the buffer level for the SBA algorithm varies more than that for the CSD algorithm. This is due to that the SBA algorithm schedules in order to meet the *delay* requirements, while utilizing the channel resources efficiently, whereas the CSD algorithm is more concerned with keeping the channel weighted buffer levels rather equal. The buffer level will therefore vary as the admitted rate varies.

■

Furthermore, we observe that the delays are kept more equal with the SBA algorithm.

Observation 8.14: *Delay target is mostly met*

In Figure 8.28 we see that the target delay for the clients, when using the SBA algorithm for scheduling, is met to a large extent. The remaining variability for e.g. clients 1 and 3 (blue and red curves respectively), is most probably due to transient effects when their admitted rates change.

■

We find SBA to be a promising algorithm that we will evaluate further in the case study in Chapter 9.

8.5 Summary

We have demonstrated that algorithms based on the quadratic criterion, as well as the probabilistic approach, possess desirable properties, especially in combination with a working admission control.

- The MAXR algorithm, which is the least complex of all the presented algorithms, might require an additional re-distribution, in order to avoid resource waste in some rare cases.
- The ITER algorithm shows almost identical performance as the CSD algorithm, but at a significantly lower complexity.
- The SBA algorithm shows desirable properties in terms of explicit delay fulfillment.

All schedulers will require an admission control algorithm to regulate the load according to SLA price models and the different clients' cost/resource efficiencies. Both the PID SLC and the LP SLC have shown interesting performance properties, in terms of adapting the load to the underlying channel properties, in the light of the SLA price models. However, it must be stressed again that the tuning of the PID parameters requires special attention in order to obtain a PID controller that handles the SLC well. We have iteratively improved the control gain K, in order to tune the PID to the different control problems,

- SLC for controlling buffer levels ($K = 0.004$),
- SLC for controlling relative buffer levels ($K = 0.4$), and
- SLC for controlling failure probabilities ($K = 0.4$).

Furthermore, we have demonstrated that by reserving a smaller portion of the resources to clients that do not accept rate variations, we can more easily meet the requirements from them. This was discussed already in Example 2.6, when introducing the service level agreements, in Section 2.3.2. Services, posing strict requirements on both throughput and delay, should be regarded as costly, and therefore be priced accordingly.

In the next chapter we will elaborate on the issues presented here, and also run longer simulations, using the emulated channels from the ray tracing models described in Section 8.2.4.

Chapter 9

Case Study

In the Wireless IP project, a system proposal for future mobile communication services, has been developed, as outlined in Section 4.3.5. The case study in this chapter focuses on the downlink of the system, which is flexible and well suited for implementation of the services described in this thesis. By this case study, we will shed some light on the possible shortcomings and advantages of applying different scheduling and service level control algorithms, to this type of system.

We will present the performance of two scheduling algorithms in combination with two service level control algorithms. These are:

- PID service level control (SLC) with ITER scheduling, in Section 9.3
- PID SLC with SBA scheduling, in Section 9.4
- Linear program (LP) SLC with SBA scheduling, in Section 9.5
- LP SLC with ITER scheduling, in Section 9.6

The choice of these combinations is motivated by the fact that the algorithms are completely different in their approach, both for SLC and for scheduling. ITER here represents the quadratic scheduling criterion, whereas SBA represents the probability based scheduling criterion. PID represents linear feedback service level control, while LP represents the mathematical program based service level control.

We begin by outlining the underlying assumptions in the case study simulations in Section 9.1 and Section 9.2. Then we perform the simulations for each case in Sections 9.3 to 9.6, before we conclude the case study in Section 9.7.

9.1 Mobile Environment

In the simulations we will use the emulated channels described in Section 8.2.4 with the parameters given in Table 8.2 on page 136. Furthermore, the assumptions outlined in Section 8.1, apply also to the simulations in this case study.

We have chosen to simulate a case with 13 simultaneous users,[1] that will be uniformly distributed in space, leading to the average SNR distribution according to Equation (8.2) due to path loss. For 13 users, the average SNR values (and their relative distances) are given in Table 9.1, and the values are also presented in Figure 9.1.

u	**1**	**2**	**3**	**4**	**5**	**6**
$l(u)$	1	3.6	4.7	5.5	6.2	6.8
$\bar{\Gamma}(u)$	40	20.5	16.6	14.1	12.3	10.8

7	**8**	**9**	**10**	**11**	**12**	**13**
7.4	7.9	8.4	8.8	9.2	9.6	10.0
9.7	8.6	7.7	7.0	6.2	5.6	5

Table 9.1: Relative distance, $l(u)$, and SNR distribution, $\bar{\Gamma}(u)$, for 13 users.

Combining the average SNRs from Table 9.1 with the small-scale fading variations from the emulated channels presented in Figure 8.1, will result in 13 different channel traces that we may use as model for the mobile environment. The resulting channels are presented in Figure 9.2, that corresponds to the data presented in Figure 8.1, but with the average values added to it. In the presented case study, we have applied channel number 1 to user number 1, and so on, leading to channel number 1 being applied to the closest user, and channel number 13 to the farthest user.

9.2 Service Levels

The QoS parameters for the different users are displayed in Table 9.2. In order to incur some change on the simulated system, apart from the channel variations, each of the "fixed rate" service users will demand a doubling of their transmission rate somewhere along the simulated time. In the simulations presented here, every 2000th time slot, one "fixed rate" user will demand a doubling of its admitted rate.

[1] Adding more users would not reveal significant new features.

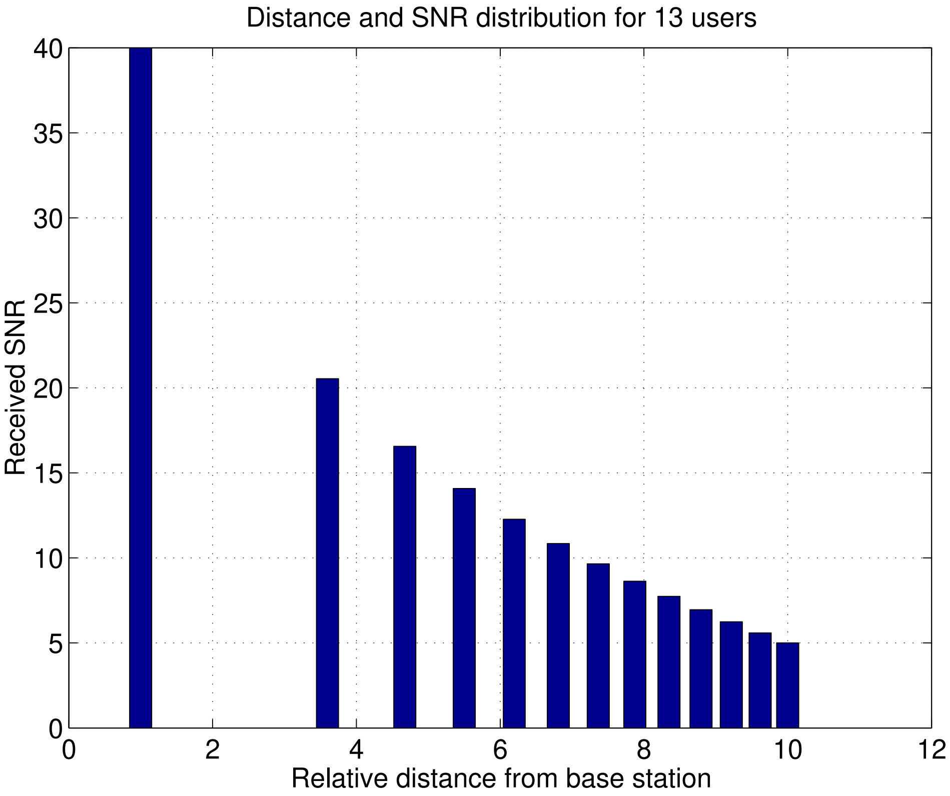


Figure 9.1: The relative distance for 13 mobile terminals from the base station, and the resulting received average SNR in dB. The mobile receiver that is closest to the base station is constrained to have a received power of 40 dB, whereas it follows that the mobile that is 10 times farther away will have a received power of 5 dB, given a path loss factor of $\alpha = 3.5$ in Equation (8.2) on page 132.

Example 9.1: Translation from MB/s to BPSK bins/time slot

In the studied system we have a time-frequency bin structure with a subdivision of the 5 MHz frequency band into 25 frequency bins, each containing 20 subcarriers. Time is divided into bins of 6 symbols each, with a period of 0.667 ms. Thus, each time-frequency bin contains 120 symbols, out of which we use 108 symbols for payload data, whereas 12 symbols are used

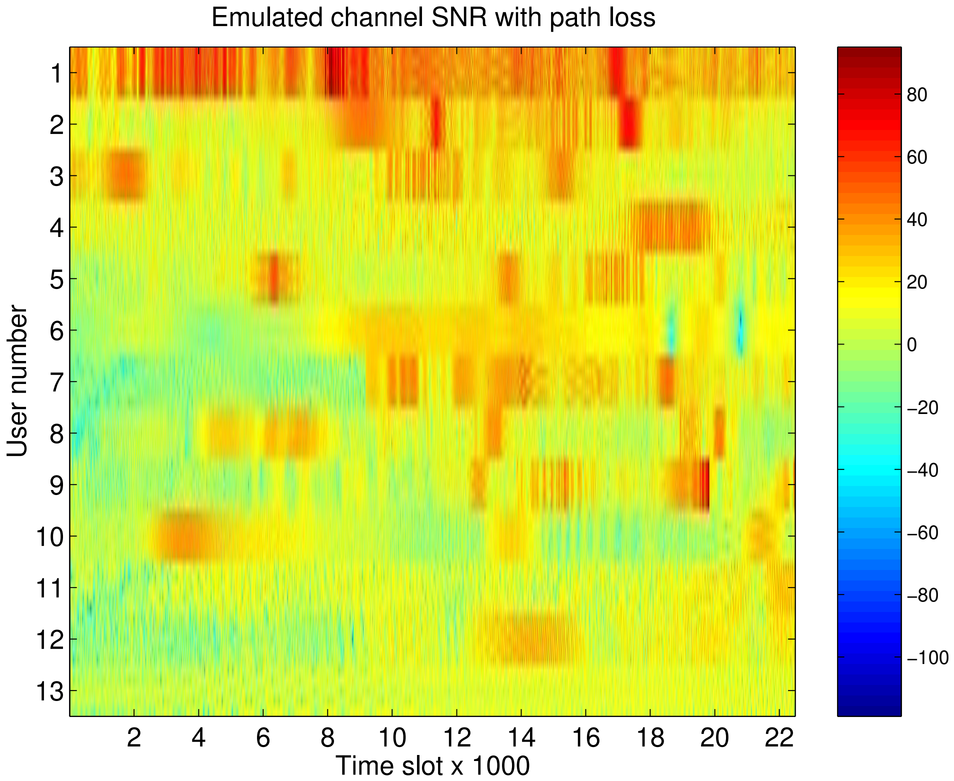


Figure 9.2: The 13 emulated channels with the imposed average SNR due to path loss. The image is presented with lower resolution in both time and frequency than the data itself.

for control signalling and pilots. Since

$$
\begin{aligned}
1\text{MB/s} &= 1 \cdot 1024^2 \cdot 8 \text{ bits/s} && (9.1)\\
&= \frac{1 \cdot 1024^2 \cdot 8}{108} \cdot 0.667 \cdot 10^{-3} \text{ BPSK bins/time slot} && (9.2)\\
&= 51.81 \text{ BPSK bins/time slot,} && (9.3)
\end{aligned}
$$

an admitted rate of 1 MB/s to one session, implies that its output buffer will be filled with 51.81 data units, corresponding to 51.81 BPSK bins, in each scheduling round.

User	Min rate	Max rate	Min price	Max price
1	0.01 MB/s	1 MB/s	0.01	0.4
2	0.05 MB/s	0.05 MB/s	0.2	0.2
3	0.05 MB/s	0.05 MB/s	0.2	0.2
4	0.05 MB/s	0.05 MB/s	0.2	0.2
5	0.05 MB/s	0.05 MB/s	0.2	0.2
6	0.01 MB/s	1 MB/s	0.01	0.6
7	0.05 MB/s	0.05 MB/s	0.2	0.2
8	0.05 MB/s	0.05 MB/s	0.2	0.2
9	0.05 MB/s	0.05 MB/s	0.2	0.2
10	0.05 MB/s	0.05 MB/s	0.2	0.2
11	0.01 MB/s	1 MB/s	0.01	0.6
12	0.05 MB/s	0.05 MB/s	0.2	0.2
13	0.05 MB/s	0.05 MB/s	0.2	0.2

Table 9.2: Service level parameters for the 13 users. Users 1, 6, and 11 are regarded as "best effort" users, for which the service level controller may adapt the admitted rates. The other users may be regarded as belonging to a "fixed rate" service. They only accept one given admitted transmission rate, and if it is not provided, a substantial economical compensation will have to be credited to their accounts.

9.3 ITER Scheduling and PID-based Service Level Control

The first case we will study consists of a PID controller, controlling the service level, and the ITER algorithm performing the scheduling. This is the least complex of all four alternative cases that we will study in this chapter. Therefore, we hope to find that its performance is comparable with other, slightly more complex, implementations.

9.3.1 PID controller

In order to provide a manageable load for the scheduler, we let a service level controller (SLC), that is part of the admission control (AC), monitor and adjust the load for the scheduler. When the channel qualities allow, then the SLC will increase the throughput requirements within the limits set by the admission policy (AP) entity, see Chapter 3. Conversely, when the channel qualities deteriorate, the SLC will reduce the throughput requirements.

In this case study, we let *one* scalar PID controller, as described in Example 3.6 on page 49, handle the buffer monitoring and adjustment of *all* the admitted data rates, through client dependent adaptive gains, K_u. It thus works as a single-input, multiple-output (SIMO) controller.

The PID controller is consulted every 40:th time-slot, that is, the admitted rates are adjusted every $40 \cdot 0.667$ ms $= 26.68$ ms. However, we will explore the impact of the choice of SLC interval in Section 9.3.6.

Example 9.2: PID parameters

A PID controller is characterized by a number of parameters that have to be tuned for the PID controller to work efficiently. They have been presented in Definition 3.3 on page 47. In the present case study we have set the values of the parameters as follows: The control gains for the different components in the PID expression from Example 3.6, K_u, K_I, and K_D are set to

$$K_u = 0.004 \frac{\sqrt{E_u^{\mathrm{sign}(e)}}}{U_r},$$

where U_r is the number of controlled sessions, E_u is the cost efficiency defined in Definition 3.4 on page 49, and e is the control error,

$$K_I = 1,$$

and

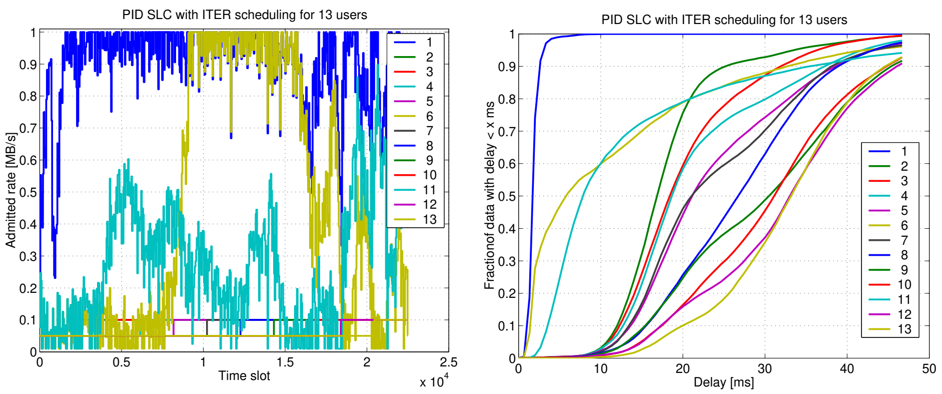
$$K_D = 0.$$

The choices of K_I and K_D are the standard initial parameters before tuning, whereas $K = 0.004$ was found after tuning in order to achieve a controller yielding smooth, but not too slow, control signals. There is thus room for additional tuning.

Recall from (3.7) and (3.8), that the price model is an integral part of E_u and will thus affect the gain of the PID. Note also that in the current setup, according to Table 9.2, the SLC can only adjust the rates of clients 1, 6, and 11.

9.3.2 ITER Scheduling Controlling Buffer Levels

In this section, we conduct three simulations using the parameters presented above, but we (or rather the admission policy (AP), see Chapter 3) vary the target buffer levels for the PID regulators, thereby primarily affecting the resulting delay, but also the total throughput. We use the emulated channels with the imposed average SNR due to path loss, according to Table 9.1 and Figure 9.2. The results are presented in Figure 9.3 to Figure 9.5. In Figures



(a) Token rates (b) Delay distribution

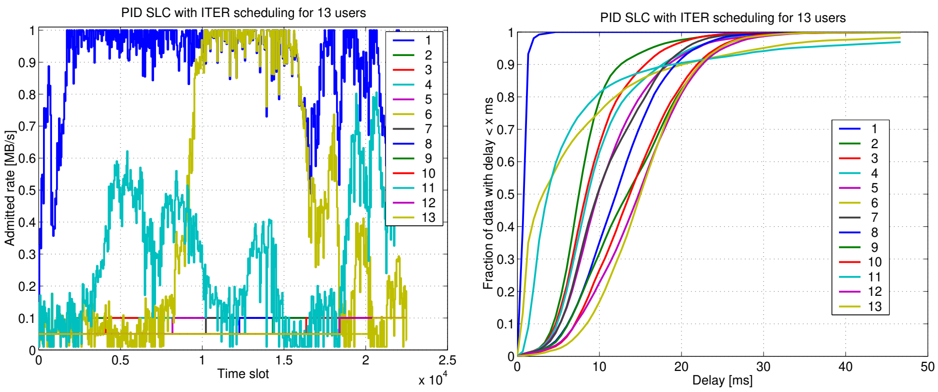
Figure 9.3: Token rates and delay distributions as controlled by the service level controller PID. Token rates are represented in MB/s, and delay distributions in fractions of the total amount of transmitted data. The reference buffer level is 200 BPSK bins. The buffers are drained according to the ITER algorithm, resulting in an accumulated throughput of 35.38 MB during 15 seconds (2.36 MB/s).

9.3(b), 9.4(b), and 9.5(b), we can see how the delay distributions vary as we let the AP vary the target buffer levels between the different simulations. In Figure 9.3(b), we have let the AP set the buffer level target to 200 BPSK bins, which is twice as much as for the case in Figure 9.4(b), which in turn is twice as much as that in Figure 9.5(b).

Observation 9.1: *Less queueing delay for smaller buffer target*

As the buffer level targets are reduced, the queueing delays become smaller. This makes sense, since there is less data waiting between having been admitted and being transmitted, when having smaller buffer targets, than when having larger buffer targets. If we look at the 80% level in the three figures, that is where 80% of the transmitted data experiences less delay than x ms, then we can see that the corresponding delay varies proportionally to the buffer level target. It is

- 40 ms for buffer level target 200 (Figure 9.3(b)),



(a) Token rates

(b) Delay distribution

Figure 9.4: Token rates and delay distributions as controlled by the service level controller PID. Token rates are represented in MB/s, and delay distributions in fractions of the total amount of transmitted data. The reference buffer level is 100 BPSK bins. The buffers are drained according to the ITER algorithm, resulting in an accumulated throughput of 35.25 MB during 15 seconds (2.35 MB/s).

- 20 ms for buffer level target 100 (Figure 9.4(b)), and
- 10 ms for buffer level target 50 (Figure 9.5(b)).

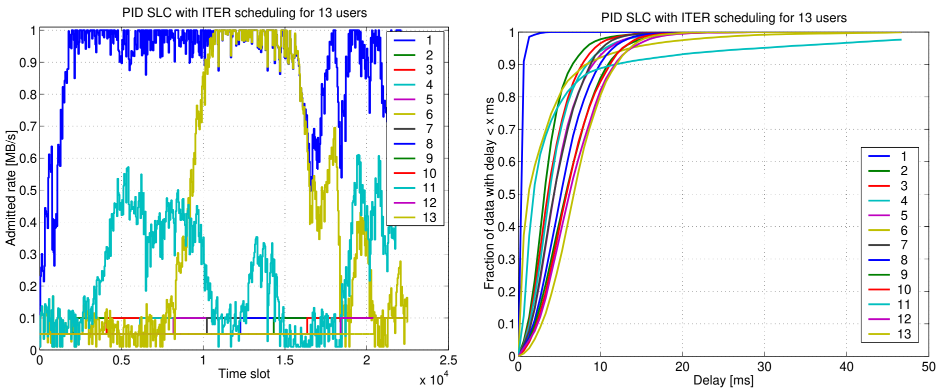
■

In order push the performance of the system further, we have run simulations also for target buffer levels of 25, 12, 6, and 3, resulting in the delays and throughputs presented in Table 9.3.

$\hat{r}$	[BPSK bins]	200	100	50	25	12	6	3
$d_{80\%}$	[ms]	40	20	10	5	2.4	1.2	0.6
$\bar{C}$	[MB/s]	2.36	2.35	2.31	2.24	2.10	1.95	1.79

Table 9.3: Target buffer level, 80-percentile delay for worst user, and total average achieved throughput. The values are also presented in Figure 9.6(a) and Figure 9.6(b), below.

We have also in Figure 9.6(a) and Figure 9.6(b) plotted the data from Table 9.3. They show the total average achieved throughput (summed over all clients) over the 15 seconds that the simulation lasted, versus the delay



(a) Token rates (b) Delay distribution

Figure 9.5: Token rates and delay distributions as controlled by the service level controller PID. Token rates are represented in MB/s, and delay distributions in fractions of the total amount of transmitted data. The reference buffer level is 50 BPSK bins. The buffers are drained according to the ITER algorithm, resulting in an accumulated throughput of 34.61 MB during 15 seconds (2.31 MB/s).

for 80% of the data for the worst user. This is summarized in Observation 9.2.

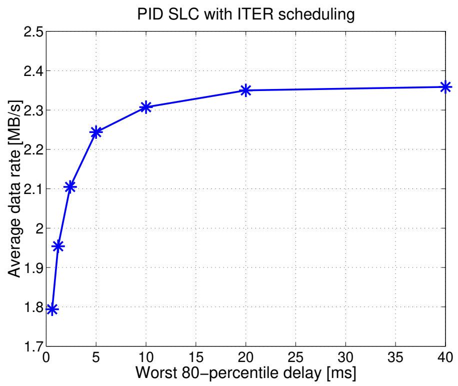
Observation 9.2: *Lower throughput for smaller buffer target*

The delay reduction discussed in Observation 9.1 comes at a certain throughput cost. The throughput versus delay plots in Figure 9.6(a) and Figure 9.6(b), indicate that there is a throughput-delay trade-off.

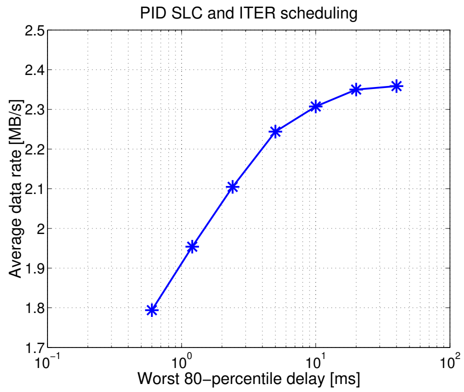
■

The purpose of having buffers at all is to give the scheduler an opportunity to behave opportunistically, and select for transmission the data flows that currently have good channel conditions. This opportunistic behaviour requires that the different data flows have data "standing by", ready for transmission, as we have in the output buffers or queues.

Having more data than 200 BPSK bins standing by for each client, in each scheduling round (time slot), is in our system pointless, since that is what we maximally can transmit during one time slot (25 frequency bins with 8-bit (256-QAM) modulation). Therefore, *target buffer levels* larger than that should be avoided. Having a larger number of active clients, the target buffer levels can be reduced further. However, larger buffers may be



(a) Throughput versus linear delay



(b) Throughput versus logaritmic delay

Figure 9.6: Average throughput $\bar{C}$ versus the maximum delay for 80% of the data for the worst client $d_{80\%}$, 9.6(a) with linear scale, and 9.6(b) with logaritmic scale on the delay.

required for the purpose of traffic shaping.

We would expect that less data in the "stand by" position would lead to less possibilities for the scheduler to exploit the transmission opportunities. We can see that when reducing the target buffer level from 200 to 50 BPSK bins, and thereby reducing the maximum delay for 80% of the data with a factor of 4, we only reduce the achieved throughput with 2%, from 35.38 MB to 34.61 MB. However, reducing it even more, such as to 3 BPSK bins, results in a throughput reduction of 24%. From Observation 9.2 and the discussion above, we conclude that we should avoid filling up the buffers more than necessary.

Observation 9.3: *Lower admitted rate fluctuations for smaller buffer target*

In Figures 9.3(a), 9.4(a), and 9.5(a), we can see how the service level controller adapts the admitted rates according to the channel variations. Studying the admitted rate variations for the controlled users (1, 6, and 11), we see that for smaller buffer target levels, we obtain smaller variations in the control signals from the PID controller, which is due to the consequently smaller error signals, e, in (3.2).

■

In order to avoid too slow control actions from the PID controller, it is thus necessary to adjust the control gain, K in (3.2), should the buffer level targets change much.

Observation 9.4: *Delay variations among fixed rate service users*
In Figures 9.3(b), 9.4(b), and 9.5(b), it is clearly the case that, for the "fixed rate" service users, the user with the largest delay has approximately twice the delay of the one with the smallest delay. It is also clear that it is always client 13 that has the largest delay, and client 2 that has the smallest delay, among these clients, in all three cases. This is due to that client 2 doubles its admitted rate early in the simulation, while client 13 does that almost in the end. Since all clients have approximately the same output buffer level, this would lead to client 2 obtaining less queueing delay, due to its higher service rate.

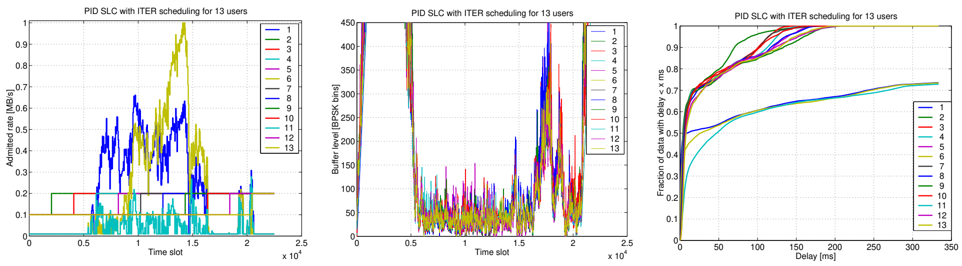
■

Observation 9.4 points at a drawback of controlling the buffer levels, without taking the admitted rates into account, thereby controlling the delays. In Section 9.3.5 we will introduce the external priority in order to indirectly control the delays, by individually adjusting the target buffer levels, and the weighting factor P_u. However, if there are no explicit delay requirements other than "as small as possible", then the network operator can in similar cases allow itself to use target buffer levels of about 25-50 BPSK bins, without significant loss of throughput, in order to provide a low scheduling latency.

9.3.3 ITER Scheduling with Saturated Service Level Control

In the following simulation we illustrate what happens when the admission control has admitted too many users or given them too high transmission rates, as compared to what the channels can handle. The service level parameters are basically the same as for the simulations in Section 9.3.2, except that for the "fixed rate" service users we have doubled their throughput requirements from 0.05 MB/sec (410 kbps) to 0.1 MB/sec (819 kbps). Furthermore, every 2000th time slot, one of them will still double its rate requirement, as in the previous experiment. This will drive the PID SLC into the saturated state, as illustrated in Figure 9.7. It is worth noting the "fixed-rate" clients that sequentially double their admitted rates from 0.1 MB/s to 0.2 MB/s, every 2000:th time-slot, in the lower part of Figure 9.7(a).

Observation 9.5: *Indefinite delays when admission control fails*
In Figure 9.7(c) we see that the delays may grow to infinity when the rates into the output buffers are too high for the channels to handle. In this case,



(a) Token rates (b) Buffer levels (c) Delay distribution

Figure 9.7: Token rates and delay distributions as controlled by the service level controller PID. Token rates are represented in MB/s, buffer levels in BPSK bins, and delay distributions in fractions of the total amount of transmitted data. The reference buffer level is 50 BPSK bins. The buffers are drained according to the ITER algorithm, resulting in an accumulated throughput of 28.62 MB.

this is particularly clear until time slot 5000 for the controlled "best effort" service users, that reduce their input rates to the minimum (Figure 9.7(a)), but still face high buffer levels (Figure 9.7(b)), leading to large delays for their admitted data.

■

A well working admission control should avoid this problem by handing off, or dropping, some clients, depending on the SLA pricing models and the channel qualities that different clients experience. The problem of *queue stability* is discussed further in Appendix B.

Observation 9.6: *Throughput reduction caused by a low rate flexibility*

In Figure 9.7 we see that the total accumulated data throughput is 28.62 MB during 15 s, which is 17% less than the comparable case in Figure 9.5. The throughput reduction makes sense, since by "reserving" a large portion of the resources for the "fixed rate" service users, we have reduced the possibility for the "best effort" service to exploit the long term channel variations with the service level control.

■

9.3.4 Controlling Buffers with More Flexible Clients

In this section we will replace some of the "fixed rate" service users with "best effort" service users, in order to increase flexibility for the admission controller's service level control entity. We hope to see that allowing more flexibility will increase the total throughput. In Table 9.4 we present the current service level parameters in this simulation setup. The target buffer level for the SLC is set to 50 BPSK bins[2]. The results are presented in Figures 9.8(a) to 9.8(b), and discussed in Observation 9.7.

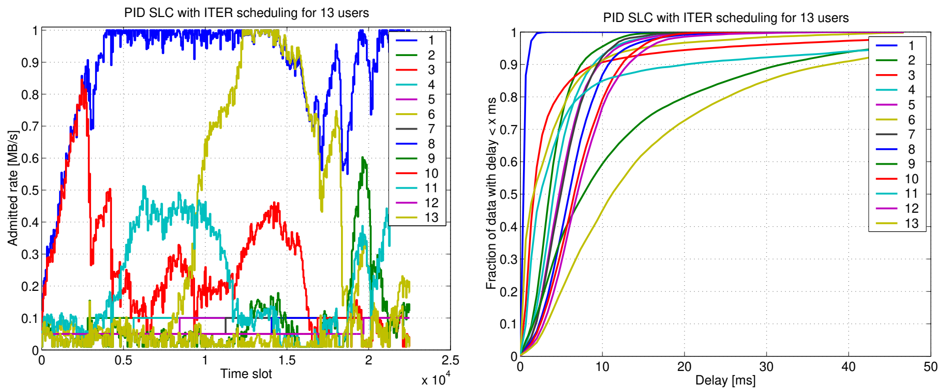
User	Min rate	Max rate	Min price	Max price
1	0.01 MB/s	1 MB/s	0.01	0.4
2	0.05 MB/s	0.05 MB/s	0.2	0.2
3	0.01 MB/s	1 MB/s	0.01	0.4
4	0.05 MB/s	0.05 MB/s	0.2	0.2
5	0.05 MB/s	0.05 MB/s	0.2	0.2
6	0.01 MB/s	1 MB/s	0.01	0.6
7	0.05 MB/s	0.05 MB/s	0.2	0.2
8	0.05 MB/s	0.05 MB/s	0.2	0.2
9	0.01 MB/s	1 MB/s	0.01	0.4
10	0.05 MB/s	0.05 MB/s	0.2	0.2
11	0.01 MB/s	1 MB/s	0.01	0.6
12	0.05 MB/s	0.05 MB/s	0.2	0.2
13	0.01 MB/s	1 MB/s	0.01	0.4

Table 9.4: Service level parameters for the 13 users. Users 1, 3, 6, 9, 11, and 13 are here regarded as "best effort" users, for which the service level controller may adapt the admitted rates. The other users may be regarded as belonging to a "fixed rate" service. They only accept one given admitted transmission rate, and if it is not provided, a substantial economical compensation will have to be credited to their accounts.

Observation 9.7: *Higher data throughput with more "best effort" users*

Comparing Figure 9.8 to Figure 9.5, that has the same target buffer level, we see that by making a larger number of the active sessions "best effort", we can increase the total data throughput. The resulting throughput of 35.49 MB is even higher than for the case in Figure 9.3, when we had four times larger buffer target, and thus more flexibility for the scheduler, at the

[2]Here, all clients have the same target buffer level, but they may as well have different target levels, which will be the case in Section 9.3.5.



(a) Token rates (b) Delay distribution

Figure 9.8: Token rates and delay distributions as controlled by the service level controller PID. In this simulation, six of the clients are "best effort" (1, 3, 6, 9, 11, and 13). Token rates are represented in MB/s, and delay distributions in fractions of the total amount of transmitted data. The reference buffer level is 50 BPSK bins. The buffers are drained according to the ITER algorithm, resulting in an accumulated throughput of 35.49 MB, during 15 seconds.

expense of a higher delay. The substantial gain is due to the possibility for the SLC to reduce the rates of users 9 and 13, that have rather bad channel conditions throughout the simulation (except for user 9 around time slot 20000, as can be seen in Figure 9.8(a)), and that user 3 can be allowed to increase its rate due to favourable channel conditions.

■

9.3.5 ITER Scheduling Controlling Delays

It is possible to control the delays explicitly, by means of setting different external priorities to different clients. However, a requirement is that the admission control does a good job in adjusting the work load for the scheduler. In the following simulation we demonstrate this by setting the target delay to 20 time-slots (13.3 ms), and then setting the external priority P_{ue} according to (6.6) and (6.7).

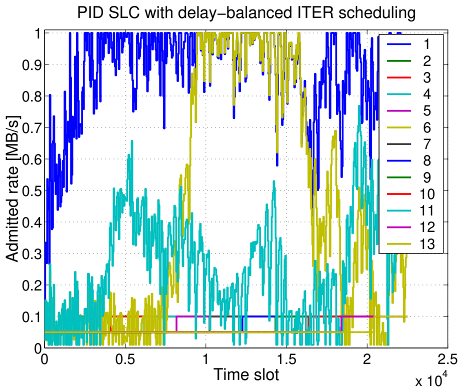
The error signal (3.3) from Definition 3.3 is calculated from the feedback signal y, which we here set to

$$y = \max_u \frac{r_u}{\hat{r}_u},$$

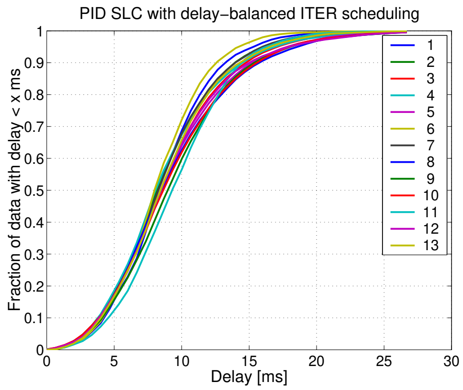
where $\hat{r}_u$ is the target buffer level calculated from Little's formula in Equation (6.6), and r_u is the buffer level for client u. This signal is subtracted from the reference signal $q = 1$ (if $\hat{r}_u = r_u$, then $y_u = 1$). Thus,

$$e(t) = 1 - \max_u \frac{r_u}{\hat{r}_u}. \tag{9.4}$$

We present the results obtained using (9.4) in the PID regulator, and $\hat{r}_u = 20$ BPSK bins, with the other simulation parameters being identical to those in Section 9.3.2, in Figure 9.9.



(a) Token rates



(b) Delay distribution

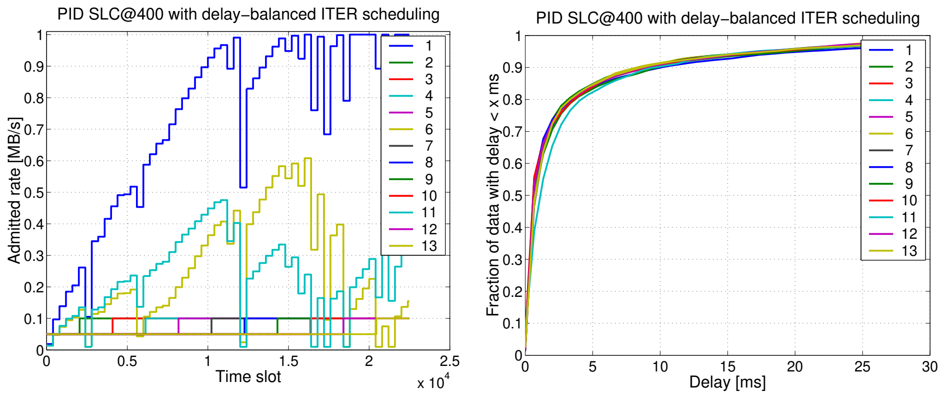
Figure 9.9: Token rates and delay distributions as controlled by the service level controller PID. Token rates are represented in MB/s, and delay distributions in fractions of the total amount of transmitted data. The buffers are drained according to the delay-balanced ITER algorithm, by setting the external priority P_{ue} according to (6.6) and (6.7), resulting in an accumulated throughput of 34.79 MB during 15 seconds. The target delay is 13.3 ms.

Obviously, it is possible to indirectly control delays with the ITER algorithm, by means of assigning rate-dependent individual

- external priorities P_{ue} for the scheduler, and
- buffer level targets $\hat{r}_u$ for the SLC.

9.3.6 PID SLC with Slower Sampling

It may be required to slow down the rate adaptation of the admission control, in order for the communicating applications to cope with the changes. We have here run a simulation where we have increased the sampling interval for the PID SLC from 26.68 ms to 266.8 ms, which corresponds to a relatively slow round-trip time for a TCP data packet. The control gain K, see Example 3.6 on page 49, has been adjusted to become slightly more careful, by reducing it from $K = 0.4$ in Section 9.3.5 to $K = 0.1$, since we have reduced the target delay to 10 time slots (6.67 ms). This reduction is required as compensation for the lower target buffer level, $\hat{r}_u$, that in Equation (9.4) will affect the error signal, e, as a higher gain on the feedback signal $y = \max_u \frac{r_u}{\hat{r}_u}$. All other service requirements are the same as in Table 9.2. We see the results from this simulation in Figure 9.10.



(a) Token rates (b) Delay distribution

Figure 9.10: Token rates and delay distributions as controlled by the service level controller PID, now updating the rates only every 400:th time-slot. Token rates are represented in MB/s, and delay distributions in fractions of the total amount of transmitted data. The buffers are drained according to the delay-balanced ITER algorithm, by setting the external priority P_{ue} according to (6.6) and (6.7), resulting in an accumulated throughput of 29.16 MB during 15 seconds. The target delay is 6.67 ms, however, many clients experience larger delays on a large portion of their data, since the fading channels vary faster than the SLC does. Clearly, the buffer levels grow large when the SLC is too slow.

Observation 9.8: *Too slow SLC increases delays*

The result of having a very slow admission control is that the buffer levels may grow between the rate updates, and thus also the delays, since the channel properties vary faster than the assigned rates. For delay sensitive clients, this may be unacceptable. It is clear that the SLC must be updated faster than every 300 ms, when using PID SLC and ITER scheduling in combination with delay-sensitive clients.

■

When used in combination with ITER scheduling, the SLC should be updated at the rate of the slow channel variations (the large scale, or shadow fading), to fully exploit these variations, but also to avoid increased delays due to mismatching resource demand and availability. In Section 9.4.3 we will explore how the SBA scheduling algorithm handles longer update intervals.

9.3.7 Conclusions from ITER Scheduling with PID SLC

We have seen that delay requirements can be supported, by means of controlling the individual buffer levels. This requires that the SLC works at a rather high update rate, perhaps higher than some communicating applications, and their protocols, may be able to handle, in order not to starve clients with tight delay requirements.

9.4 SBA Scheduling and PID Service Level Control

We will here combine the SBA scheduler, described in Section 7.4.2, with the PID service level controller, described in Example 3.6 on page 49. The service level parameters are given in Table 9.2, and also in these simulations, each "fixed rate" client will double its rate every 2000:th time-slot.

The purpose is to explore what influence the target failure probability $\hat{P}_u^F$ will have on the performance, and also to see whether narrower delay requirements have a large impact on the peformance.

9.4.1 PID Controller

The SBA scheduling algorithm works with probabilities, and the PID SLC will regulate the admitted rates with respect to the failure probability, P_u^F. The PID controller will be almost identical to the one described in Example 9.2 in Section 9.3.1, except for the gain factor K, that here is set to

$K = 0.4$, yielding

$$K_u = 0.4\frac{\sqrt{E_u^{\mathrm{sign}(e)}}}{U_r},$$

where E_u is the cost efficiency from (3.7) and (3.8), e is the control error, and U_r is the number of controlled clients. The gain K is increased as compared to the PID gain used in combination with the ITER scheduling algorithm, since the feedback signal y, and the reference signal q, used for calculation of the control error, $e = q - y$, are here the remaining maximum failure probability $y = \max_u P_u^F$, and the target failure probability $q = \hat{P}_u^F$, after the scheduling round. Since $0 \leq P_u^F \leq 1$, then the error will be in the interval $-1 \leq e \leq 1$, which is much less than for the ITER scheduling case described in Section 9.3.1, where the error signal is calculated directly from the buffer levels.

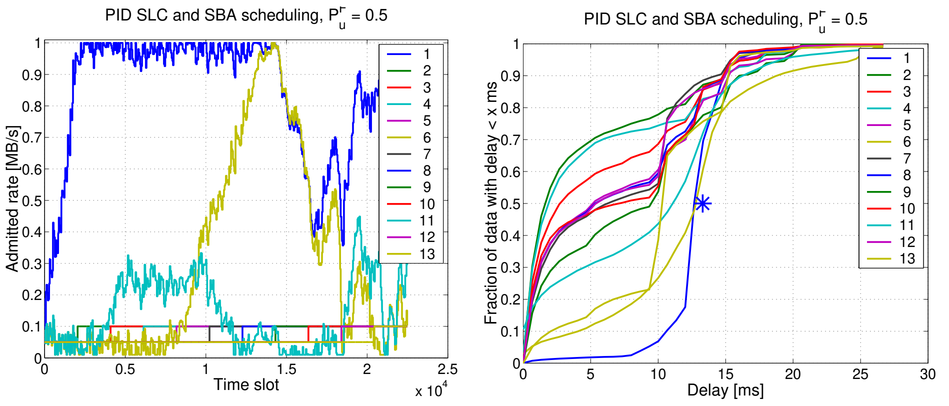
In this case we will have a common target failure probability for all clients, but it is possible to have individual targets, in order to deliver different service quality levels.

9.4.2 SBA Scheduling Controlling Delays

In Figure 9.11, we present a simulation where the target delay for the SBA scheduling algorithm was set to 20 time-slots, that is 13.3 ms. The target failure probability for the PID SLC was set to 50% in this first simulation, but in the subsequent simulations we will reduce it to 30% and 10%, respectively, to illustrate the effect of having different target failure probabilities. In Figure 9.11(a) we see that the admitted rates are rather high, but lower than for the corresponding simulation with the ITER scheduling algorithm, in Figure 9.9. However, we see a different behaviour in the delay distributions. The target delay and the failure probability is marked with a star (*) in Figure 9.11(b), and we can see how the delay distribution curves for clients 1 and 6, almost exactly meet the delay requirements (for client 1, 80% of the data has a delay in the interval $11 \leq d \leq 15$ ms).

In the simulation presented in Figure 9.12, we have kept the target delay, but changed the target failure probability from 50% to 30%. As compared to Figure 9.11, Figure 9.12 shows a significantly improved delay performance, at the expense of a slightly reduced throughput, from 30.29 MB to 29.14 MB during 15 seconds.

In this last simulation, presented in Figure 9.13, we have kept the target delay, but reduced the target failure probability even more, to 10%. For the case presented in Figure 9.13, the delay has improved further, with



(a) Token rates (b) Delay distribution

Figure 9.11: Token rates and delay distributions as controlled by the service level controller PID. Token rates are represented in MB/s, and delay distributions in fractions of the total amount of transmitted data. The buffers are drained according to the SBA algorithm, resulting in an accumulated throughput of 30.29 MB during 15 seconds. The target delay is 13.3 ms, and the target failure probability is 50%, indicated by the star (*) in Figure 9.11(b).

another slight reduction in throughput, from 29.14 MB to 27.30 MB during 15 seconds.

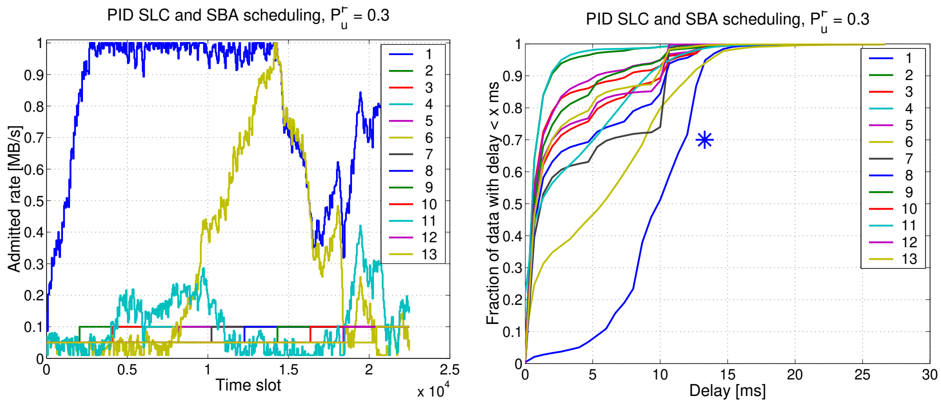
Observation 9.9: *Tighter delays give less throughput*

Again, as in the case with the ITER algorithm in Section 9.3.2, we note that tightening the delay requirements results in a reduced throughput. ■

However, it is worth noting how well the explicit delay requirements are met by the combination of a PID SLC with an SBA scheduling algorithm, in all three cases described above.

9.4.3 PID SLC with Slower Sampling

We will here explore how the SBA scheduling algorithm handles longer update intervals. From Section 9.3.6, we learned that with the ITER algorithm, the delays grew large for all clients when the channel worsened faster than the SLC could react. We hope that the SBA scheduling algorithm will not give the same result, since we observed in Observation 8.11, on page 154,



(a) Token rates (b) Delay distribution

Figure 9.12: Token rates and delay distributions as controlled by the service level controller PID. Token rates are represented in MB/s, and delay distributions in fractions of the total amount of transmitted data. The buffers are drained according to the SBA algorithm, resulting in an accumulated throughput of 29.14 MB during 15 seconds. The target delay is 13.3 ms, and the target failure probability is 30%, indicated by the star (*) in Figure 9.12(b).

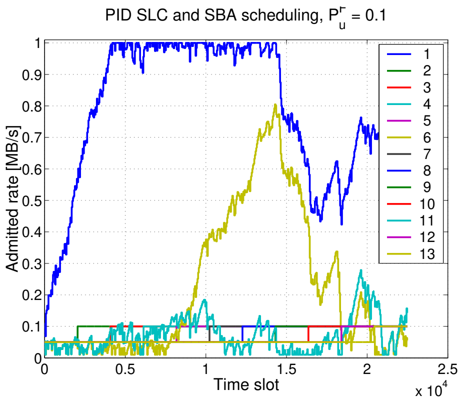
that the SBA algorithm tends to "give up" on clients that are assigned too high rates, thereby saving more resources to the other clients. We present the results in Figure 9.14.

In Figure 9.14 we see how the admitted rates fail to reach the same levels as in the previous simulations. The rate changes also show a slightly oscillative behaviour with large increments followed by smaller decrements, indicating that the PID controller outputs slightly too large control increments. However, the delay requirements are well met by the scheduler, in the sense that the actual failure probabilities are near the target.

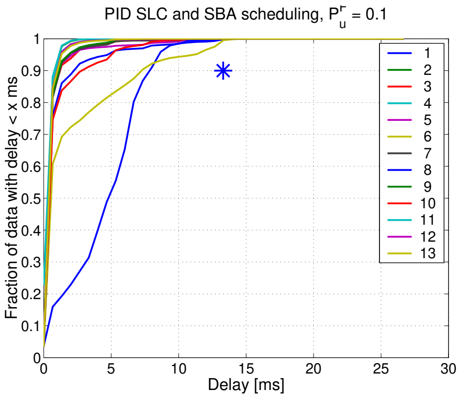
Observation 9.10: *Failure probability still high*

As seen in Figure 9.14(b), the delays are still large for a substantial fraction of the transmitted data. Furthermore, the resulting throughput of 27.3 MB during 15 seconds is even less than that for the ITER algorithm in Section 9.3.6, therefore failing to outperform the ITER algorithm in terms of delay and throughput.

■



(a) Token rates



(b) Delay distribution

Figure 9.13: Token rates and delay distributions as controlled by the service level controller PID. Token rates are represented in MB/s, and delay distributions in fractions of the total amount of transmitted data. The buffers are drained according to the SBA algorithm, resulting in an accumulated throughput of 27.3 MB during 15 seconds. The target delay is 13.3 ms, and the target failure probability is 10%, indicated by the star (*) in Figure 9.13(b).

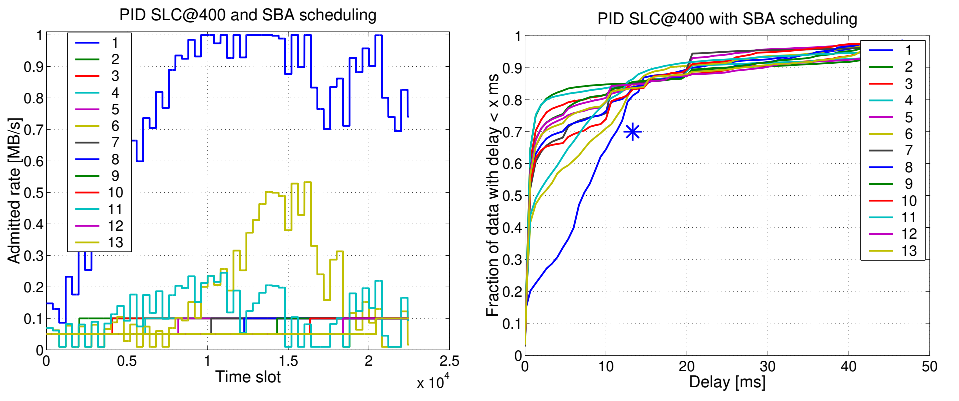
In order to see the effects of more narrow delay requirements, we alter the failure probability to be $P_u^F = 10\%$, and the target delay to be $T_u = 10$ time slots, that is 6.67 ms. The results are presented in Figure 9.15.

As we can see from Figure 9.15, the PID SLC becomes very cautious when controlling the rates, since the target failure probability is set so low. However, it is interesting to note the relatively high admitted rate assigned to client number 11, in Figure 9.15(a). It is approximately half that of the rate assigned to client number 1, resembling the case in Figure 9.10(a), but at a lower absolute rate.

Observation 9.11: *Delay requirements yield considerably reduced throughput*

Because of the low target failure probability, and the low target delay, the throughput is considerably reduced, with more than 20%. However, the delay requirements are fulfilled.

■



(a) Token rates (b) Delay distribution

Figure 9.14: Token rates and delay distributions as controlled by the service level controller PID. Token rates are represented in MB/s, and delay distributions in fractions of the total amount of transmitted data. The SLC is updated every 400:th time-slot. The buffers are drained according to the SBA algorithm, resulting in an accumulated throughput of 27.3 MB during 15 seconds. The target delay is 13.3 ms, and the target failure probability is 30%, indicated by the star (*) in Figure 9.14(b).

9.4.4 Conclusions from SBA Scheduling with PID SLC

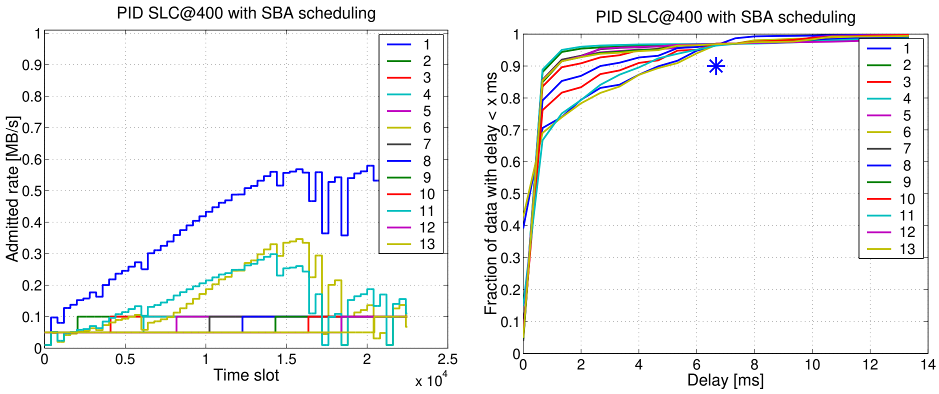
From Observations 9.10 and 9.11 we again conclude that services with tight delay requirements have to yield a higher price for the same throughput, than services with less strict delay requirements. The bottom line is that we were not helped by the change of scheduling algorithms.

9.5 SBA Scheduling with Linear Program SLC

As we pointed out in Section 3.3.2, we expect the linear program service level control to find optimal rate allocations, in terms of maximizing revenue.

9.5.1 Linear Program

In order to create a linear program that solves the same rate control problem as the previously presented PID controllers did, we need to express our price models in terms of a linear objective function (3.9) and rate limits (3.11), from page 52. Furthermore, the cost efficiency and the available resources



(a) Token rates (b) Delay distribution

Figure 9.15: Token rates and delay distributions as controlled by the service level controller PID. Token rates are represented in MB/s, and delay distributions in fractions of the total amount of transmitted data. The SLC is updated every 400:th time-slot. The target delay is 6.67 ms, and the target failure probability is 10%, indicated by the star (*) in Figure 9.15(b). The buffers are drained according to the SBA algorithm, resulting in an accumulated throughput of 21.63 MB during 15 seconds.

come in as constraints through Equation (3.10).

Example 9.3: LP formulation of price model from Table 9.2

In the objective function (3.9), the price model dictates the values of the parameters $\frac{\Delta e_u}{\Delta I_u}$, according to

$$\frac{\Delta e_u}{\Delta I_u} = \frac{\text{Price max - Price min}}{\text{Rate max - Rate min}}, \tag{9.5}$$

unless it is a fixed rate service. For fixed rate servcies, such as all clients in Table 9.2 except for clients 1, 6, and 11, the value we set in (9.5) does not matter, since the SLC cannot change their rates. Thus, for client number 1, we have

$$\frac{\Delta e_1}{\Delta I_1} = \frac{0.4 - 0.01}{1 - 0.01} = 0.3939 \text{ [Cents/(MB/s)]}$$

The upper and lower rate limits from the price model, $\overline{I}$ and $\underline{I}$, respectively, also come in through the constraints (3.11), as

$$\underline{I}_u \leq I_u \leq \overline{I}_u, \tag{9.6}$$

which in the case for client number 1 becomes (see Table 9.2)

$$0.01 \leq I_1 \leq 1,$$

and for the fixed-rate client number 2 becomes

$$0.05 \leq I_2 \leq 0.05.$$

To include also the cost of supplying the resources, in the linear program, we set the paramters in (3.10) according to the following example.

Example 9.4: Inclusion of cost and resource constraints into the LP formulation

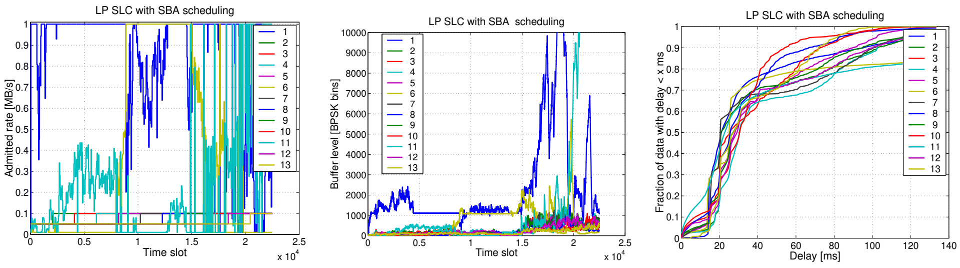
Assuming we express the admitted rates I_u in MB/s (megabytes per second), the average allocated bin capacity C_u in bits per symbol, and the available resources in symbols per second, Equation (3.10) becomes

$$\sum_{u=1}^{U} \frac{I_u}{C_u} \leq \frac{S \cdot n_s}{T_s} \cdot \frac{1}{8 \cdot 1024^2}, \tag{9.7}$$

where S is the number of available resource bins in one scheduling round, n_s is the number of payload symbols in one resource bin, and T_s is the time duration of a scheduling round (time slot). For our example system, $S = 25$ bins, $n_s = 108$ symbols , and $T_s = 0.667 \cdot 10^{-3}$ seconds.

9.5.2 LP SLC Without Buffer Level Rate Correction

In Figure 9.16 we present the result of applying the SBA scheduler to the offered load as controlled by the linear program in Section 9.5.1. Clearly, the buffer levels grow out of bounds after time slot number 15000, which also shows in the delay distribution. The reason for this growth is that client



(a) Token rates (b) Buffer levels (c) Delay distribution

Figure 9.16: Token rates, buffer levels, and delay distribution, as assigned by the Linear Program service level controller. Token rates are represented in MB/s, and delay distributions in fractions of the total amount of transmitted data. The buffers are drained according to the SBA algorithm, resulting in an accumulated throughput of 32.53 MB during 15 seconds.

number 1 meets slightly worse channels than its admission has calculated for, which results in more tokens remaining in the buffer than there would have been, had the channels been better. Since the linear program rate controller does not take this remaining level into account, the buffers will integrate up the remaining tokens with the new tokens. In order to address this issue, in the following section we make a slight correction to the decision made by the linear program.

9.5.3 LP SLC With Buffer Level Rate Correction

We will now test the same linear program as in the previous section, but we will apply a simple correction to the calculated optimal rate, in order to accomodate also the old data (or tokens) that is already waiting in the buffer for transmission, and not only the incoming data.

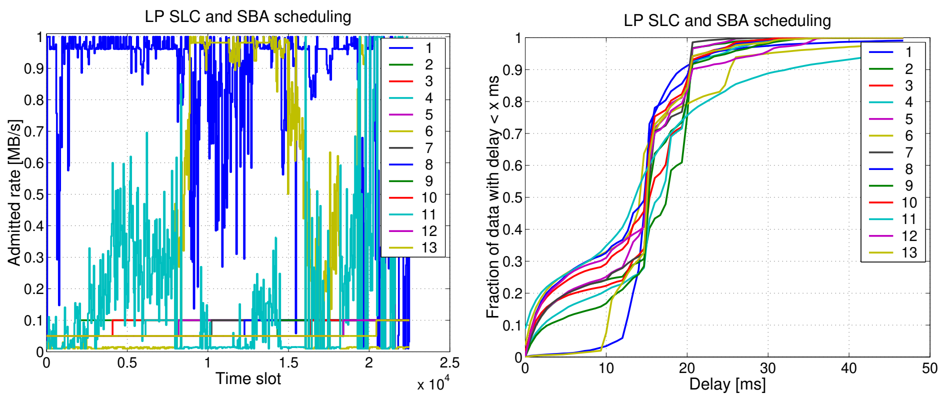
Example 9.5: Adjustment of admitted rates in order to control delay

The idea behind our modification of the LP decision is to make the LP decision "aware" of the buffers and their levels. In order to control the delay, we will have to calculate the required rate, $\hat{I}_u$, to take the buffer level from the current level, r_u, to the level $\hat{r}_u$ that corresponds to the client's delay requirements (according to Little's theorem, in (6.6)), during the time until the next SLC decision (40 time slots):

$$\hat{I}_u = \frac{\hat{r}_u - r_u}{40} \tag{9.8}$$

This required rate, $\hat{I}_u$, is subtracted from the LP decision I_u. The resulting rate $I_u - \hat{I}_u$ also has to be limited by the rate limits in Equation (3.11).

In Figure 9.17 we present the impact of this correction. Note the difference in scaling on the delay axes when comparing Figure 9.17(b) to Figure 9.16(c).



(a) Token rates (b) Delay distribution

Figure 9.17: Token rates and delay distribution, as assigned by the Linear Program service level controller. Token rates are represented in MB/s, and delay distributions in fractions of the total amount of transmitted data. The buffers are drained according to the SBA algorithm, resulting in an accumulated throughput of 32.40 MB during 15 seconds. The target delay is set to 13.3 ms in (9.8).

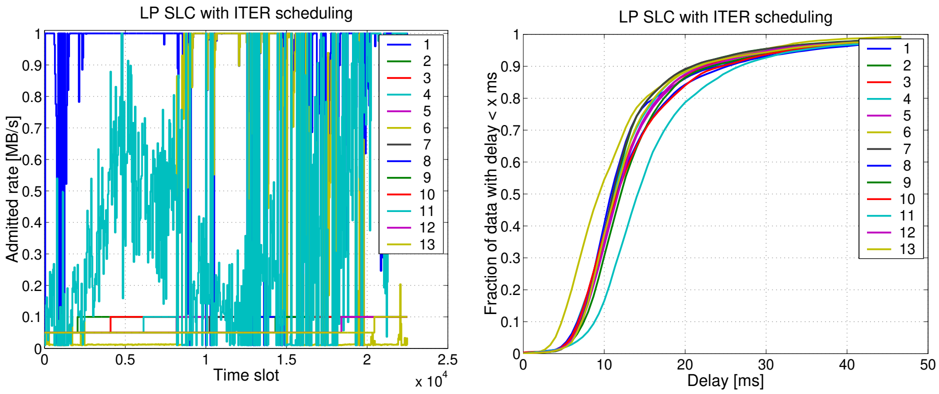
Observation 9.12: *Reduced delay and slightly reduced throughput*
In Figure 9.17 we can see, as we expected, that the buffer level rate correction helped to substantially reduce the delay for all clients. Furthermore, the effect on the total amount of transmitted data was not severe, although the correction, in most cases resulted in a reduction of the admitted data rates.

■

Apparently, SLC by a linear program works rather well with the additional buffer level dependent rate correction. This correction, or a better one[3], is required in order to uphold controlled and acceptable delays.

9.6 ITER Scheduling with Linear Program SLC

In this experiment, the linear program will be exactly the same as in Section 9.5.3. We will also apply the same buffer level dependent rate correction to the linear program solution. Note that there is no buffer-dependent feedback to the linear program, except for the post-correction of the calculated optimal rates. There is thus no need to alter any parameters in the original[4] linear problem formulation. The results of this simulation are presented in Figure 9.18.



(a) Token rates

(b) Delay distribution

Figure 9.18: Token rates and delay distributions as controlled by the linear program service level controller. Token rates are represented in MB/s, and delay distributions in fractions of the total amount of transmitted data. The SLC is updated every 40:th time-slot and the target delay is 13.3 ms. The buffers are drained according to the ITER algorithm, resulting in an accumulated throughput of 35.51 MB during 15 seconds.

[3]One could consider a non-constant correction that performs a larger rate correction for a shorter time-interval, and then returns to the LP decision.

[4]Recall that we have to change the gains in the PID SLC, depending on what type (absolute buffer level target, relative buffer level target, or, target failure probability) of buffer level target we currently work with.

From Figure 9.18(b) we can see that the delays are kept better with the ITER scheduling algorithm than with the SBA scheduling algorithm. This is also reflected in in the higher throughput obtained with the ITER algorithm, which primarily is a result of better scheduling performance, and secondarily a result of the resulting smaller buffer level rate corrections. To see this, compare the top of Figure 9.18(a) to the top of Figure 9.17(a), and note the "ripple" due to buffer level rate corrections in Figure 9.17(a), which is hardly noticeable in Figure 9.18(a).

9.7 Conclusions

Two scheduling algorithms, one based on each citerion, have been tested in combination with each of the proposed service level controllers, and evaluated in terms of throughput and delay. Results show that both schedulers can, when feasible, meet explicit throughput and delay requirements, while at the same time allowing the service level controller to maximize revenue by allocating the surplus resources to less demanding services.

The ITER scheduling algorithm, based on the quadratic criterion (6.2), was found to perform better, in terms of throughput and delay, than the SBA scheduling algorithm, which is based on the probabilistic criterion (7.15).

Clearly, as we can see in all comparable experiments, when we allow the delays to increase up to a certain limit, as exemplified in Figure 9.6(b), then we can exploit the full potential of the scheduling algorithms, and thus also maintain a high throughput.

Chapter 10

Conclusions and Future Work

10.1 Conclusions

In this thesis we have presented a novel approach to resource allocation in wireless communication systems in which economical aspects are explicitly accounted for.

Network operators now have the possibility to fully differentiate between different content providers through Service Level Agreements: Depending on the requirements on throughput and delay, different price models can be created to match the content provider's profile.

We have proposed a resource allocation structure consisting of an admission controller and a scheduler, where the admission control function is split in two parts: Admission Policy, for execution business decisions, and Service Level Control for implementing the business decisions according to the different price models and the predicted channel quality. We have shown that this structure has numerous advantages and that it manages to fulfill the users' requirements in terms of both delay and throughput while adhering to the service level agreements.

The resource allocation structure presented in this thesis is both flexible and powerful. It can be regarded as a cascade controller with the scheduler in the inner loop, the service level controller in the middle and the admission policy in the outer loop. From a control point of view this is attractive as the different loops operate on different time scales. Thus we may introduce optimisation on different levels, with revenue on the top level.

One of our main objectives has been to propose solutions that have low complexity and as such could be implemented in current systems. For the service level control we have derived a novel approach where a single PID controller governs the data flow to many users, the only difference among the users being an adaptive gain updated by cost efficiency and channel quality. We also show that service level control can be formulated as a linear programming problem. Both of these service level controllers are shown to work nicely in a detailed case study, and on real world channel data. The service level controller sets the target for the inner scheduling loop, for which different scheduling strategies have been proposed, that take both service requirements and varying channel properties into account. We have thereafter investigated several approaches, both deterministic and stochastic. Of particular interest are two novel approaches one of which is based on a quadratic criterion and the other on estimated probability distributions. Both these methods perform well in real scenarios and have an attractively low complexity.

To fully evaluate the utility of the proposed resource allocation approach a detailed simulation study has been conducted on a downlink OFDM proposal for the fourth generation communication systems. Based on these simulations we are confident that our approach to the resource allocation problem is feasible and will, if implemented, perform well in line with a wide variety of network operator objectives.

10.2 Where to go from Here

We lack a simple method for obtaining an optimal scheduling decision, given the service rate CDF and the channel capacity CDF. In the thesis we chose to use an intuitive approach, and that was to maximize the product of the two, but it may be fruitful to consider more detailed and accurate models of the overall probability of service success.

It may also be fruitful to pursue a scheduling criterion that takes into account the influences from the neighboring cells, both positive and negative. This may have consequences also on the algorithmic complexity in the scheduler, since we will have to consider the different possibilities:

- Totally non-cooperative transmissions
- Intereference reducing cooperation (silent interferer)
- Coherent combining cooperation (MISO transmission)

Spatial Diversity Scheduling

In a two-cell case, a user being on the cell boundary may receive signals from both base stations simultaneously and coherently, combining the signals into one stronger signal. This approach does not only increase the received signal power, but it also reduces the interference experienced by that user. The SNIR for a user is given by

$$\gamma_u = 10 \log \left(\frac{S_0}{N_u + \sum_{b=1}^{B} S_b} \right) \tag{10.1}$$

where B is the number of base stations, except for its home base station, that affect that user. The signal may be enhanced for that user by letting the strongest interfering base station transmit to that user coherently:

$$\gamma_u = 10 \log \left(\frac{S_0 + S_1}{N_u + \sum_{b=2}^{B} S_b} \right) \tag{10.2}$$

Taking signals from more base stations improves signal quality even more:

$$\gamma_u = 10 \log \left(\frac{\sum_{s=0}^{B_c} S_s}{N_u + \sum_{b=B_c+1}^{B} S_b} \right) \tag{10.3}$$

An improvement of less than $6dB$ per participating base station is expected from this approach. There is however a limit where capacity starts to reduce, since mobiles in a cell will lose resource slots when their home base station transmits to mobiles belonging to other base stations.

Interference scheduling

An alternative approach that may also increase signal quality for a user is by simply letting the most interfering base station be quiet, thereby improving the interference situation for that user. This approach will lead to an improvement for user u of less than $3dB$ per participating base station. However, this smaller improvement will come cheaper since there will not be a need for coherent MISO transmission at the base stations. The signal quality experienced by user u will then be

$$\gamma_u = 10 \log \left(\frac{S_0}{N_u + I_{tot} - \sum_{b=1}^{B} S_b} \right) \tag{10.4}$$

where B now is the amount of silent base stations and I_{tot} is the interference from all base stations under non-cooperative conditions.

10.2.1 Other Applications

Throughout the creation of this thesis, the author has reflected that resource allocation is not a problem only in wireless communications, or other obvious applications, such as workshop scheduling, processor sharing, etc. Society is filled with people that are working, or living, with a malfunctioning admission controller. They try to solve the problems by scheduling, but that is a dead end. If the resources are short, scheduling won't take you out of the problem. Guilty conscience will grow like buffers under bad admission control. If the load is too high, realize it, and reduce the input rate. Just say no!

Appendix A

Background Material for Reference

A.1 Link Adaptation

We here give an introduction to radio link adaptation, in order to provide a basic understanding for the potential of the flexibility that is introduced by means of channel-predictive scheduling.

A.1.1 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.

A.1.2 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 [57, 65, 86].

The impact of a temporary bad channel on the bit error rate when transmitting digital information can be seen in Figure A.1. Here, the mobile is transmitting with a constant modulation while the channel experiences several fading dips.

If we instead 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 [23, 49, 54, 82, 83], then maybe the problems can be mitigated. In Figure A.2, 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 Rayleigh fading narrow-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 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 A.3. Here we can see that the bit error rates are much lower than in Figure A.1,

[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

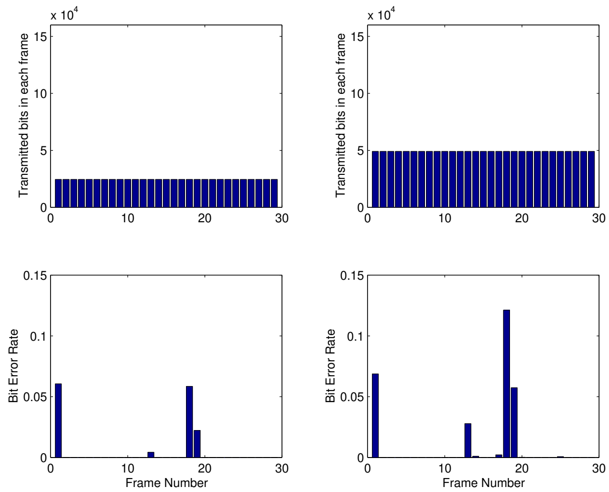


Figure A.1: Constant BPSK (left) and QPSK (right) modulation and resulting error bursts when the channel experiences fading dips, especially around frame number 18. The two top graphs show the number of transmitted bits in each time-frame, whereas the two lower graphs show the bit error rates in each frame. The corresponding channel time variability is depicted in Figure A.2, where one frame corresponds to 48 time-slots, or $5ms$.

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 [33, 40, 41]. This also gives a more fine-grained resolution of the channel capacity $C(u, s)$.

In a third step, assuming that all channels can be predicted, we could use the predictions for concurrent resource allocation in a shared physical medium. This is what is meant by the term *scheduling*, which is discussed in Chapter 4.

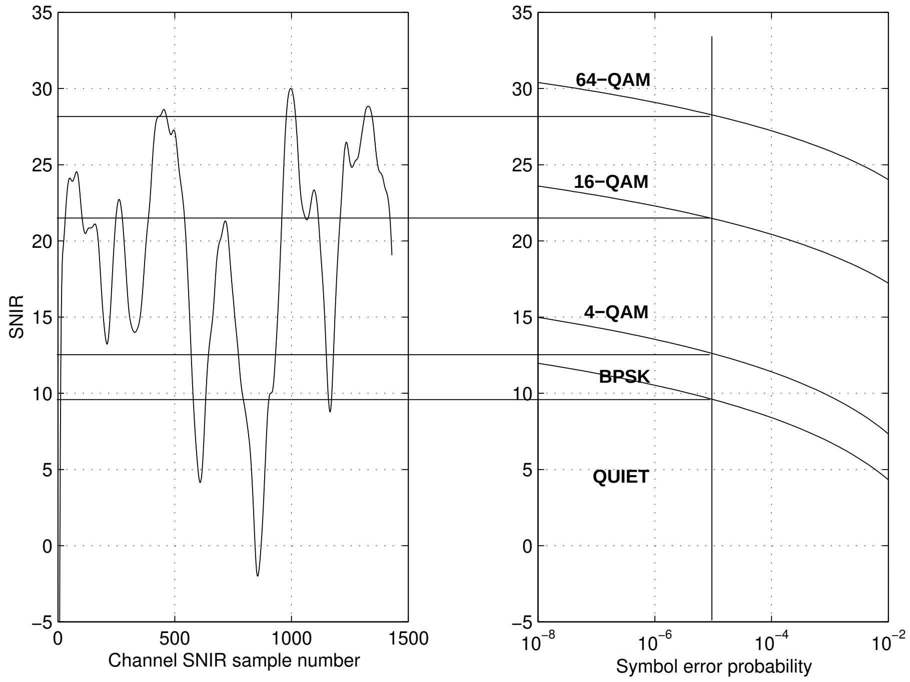


Figure A.2: SNIR trend and modulation level related to the symbol error probability. On the right hand side we see how a chosen target error rate (10^{-5}) translates into a modulation level for a given SNIR.

A.1.3 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 stack 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 [23, 49, 54, 82, 83]. In an adaptive modulation system, the symbol alphabet can be varied from one symbol to the next.

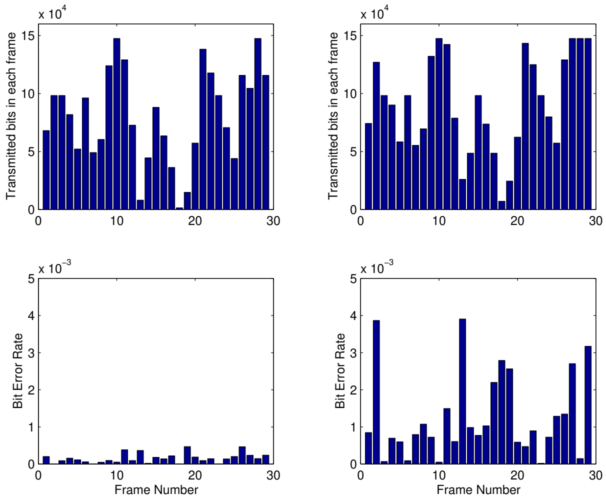


Figure A.3: The bar plots show the result of applying the same channel impairments to a mobile, using adaptive modulation with two different target symbol error rates (10^{-3} on the left hand side, and 10^{-2} on the right). The two top graphs show the resulting number of transmitted bits for each time 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 A.1, 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 A.1.

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 A.2, dictated by the application that is currently using the link. For instance, a speech application does not require as high a protection as a data file transfer.

A.1.4 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 received data chunk. 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 e.g. TCP. Therefore we could allow the link layer to have its own automatic repeat request (ARQ) system, through feedback of acknowledgment (ACK) or negative ACK (NAK) signals. So instead of just dropping an erroneous data chunk, it would send back a NAK and expect the transmitter to re-transmit the same data [46]. Alternatively, if Hybrid ARQ [36, 37] is used, an incremental piece of data which can be combined with the already received data, 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.

A.2 Robin Hood

With the Robin Hood algorithm [32], the scheduler performs the scheduling in two rounds: The first round is performed by the MAXR algorithm, described in Section 6.3.2. In the second round, resource bins are redistributed from clients that have been over-supplied (rich), to clients that have been under-supplied (poor), with respect to their admitted data. We call this equalization 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 MAXR algorithm.

2. Find the rich and poor clients by comparing their allocations to their amount of data in the queues (or admitted tokens in their buffers)

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

(a) For the richest client, find its worst resource bin, in the sense of providing the smallest contribution to its criterion.

(b) Among the poor clients, find the best client (in terms of the largest contribution to the criterion) to use the resource bin found in step 3a, and give the resource bin to her. In case two or more poor clients have the same bin criterion, choose randomly among them.

(c) Update the rich and poor variables.

A crucial requirement for the algorithm to converge is that never must any rich clients become poor, or vice versa. This is realized by removing from the re-distribution algorithm, the clients that happen to change status from rich to poor,or vice versa.

Appendix B

Analysis

B.1 Stability of the Scheduled Queueing System

In [34, 35, 78, 79], a family of wireless scheduling criteria functions, yielding stable queues, have been suggested. In [78, 79] it is argued that any scheduling criterion that maximizes a linear combination of the products of different users queue sizes and their possible transmission rates, yields a stable queueing system. Stability is expressed in terms of finite queue sizes given bounds on the traffic and channel behaviours. The authors of [34, 35] generalize this result by replacing the queue size with a non-decreasing function of the queue size, with some additional properties. In the more general case from [34, 35], the expression that shall be maximized is given by

$$\vec{\mu}[k] \in \arg\max_{\vec{\eta}\in\mathcal{S}_{s[k]}} \sum_{i=1}^{N} f_i(x_i[k])\eta_i \tag{B.1}$$

where $x_i[k]$ is the queue size for user i at time k, and η_i is the feasible transmission rate for user i. The function $f_i(x)$ is the non-decreasing function of the queue size, and $\mathcal{S}_{s[k]}$ introduces the channel constraints at time k. The criterion in (B.1) is intuitional in that it makes sense to assign a high throughput to a large queue in order to guarantee queue stability and maximize throughput simultaneously.

In this section we will compare our general criterion (6.1) with (B.1) and other similar criteria in the literature [6, 5], in order to explain why the criteria suggested in the literature enjoy the stability and throughput optimality properties.

B.1.1 Queue Stability for Linear Approximation

Recall our general criterion in (6.1). While $n > 1$, the criterion will prioritize larger queues r_u over smaller queues (if the allocation sizes a_u and the weighting factors P_u are equal for all users). Thus, the exponent n is required to be larger than 1 in order for the criterion to prioritize larger queue sizes when making a scheduling decision. What the criterion does, is that it forces the scheduler to make a decision at time t, that minimizes the resulting value of the criterion at time $t+1^-$, when the decision has been executed, but before new data has entered into the queues(thus the "$-$" superscript on 1^-). If we add a time index to (6.1) we will see this more clearly.

$$J(t) = \sum_{u=1}^{U} P_u(t)|r_u(t)|^n \tag{B.2}$$

$$J(t+1^-) = \sum_{u=1}^{U} P_u(t+1^-)|r_u(t+1^-)|^n \tag{B.3}$$

$$= \sum_{u=1}^{U} P_u(t+1^-)|r_u(t) - a_u(t)|^n \tag{B.4}$$

Thus we see that by properly choosing the allocation a_u at time t we minimize the criterion value at time $t+1^-$. Similarly to the outline in Section 6.2 for the case of $n = 2$, we now perform the linearization for the more general case $n > 1$, using the alternative approach in Section 6.2.1. We will see that this linearization leads to an expression similar to (B.1). Furthermore, we will demonstrate that solving (6.1) will yield a higher throughput and thus a larger stability region, than solving the linearized problem (B.8) below.

Starting from the expression for a general cost function

$$J = \sum_{u=1}^{U} g(r_u) \tag{B.5}$$

where we have omitted the allocation a_u, since we now regard it as a change in the buffer level r_u. Differentiate this expression with respect to r_u.

$$\frac{\delta J}{\delta r_u} = g'(r_u) \tag{B.6}$$

If infinitisimally small changes δr_u of the buffer levels could be realized, they would result in

$$\delta J = g'(r_u)\delta r_u \tag{B.7}$$

Consider a finite, realizable variation $-\Delta r_u$, that represents a decrease in buffer level due to a resource bin being assigned to client u. Here, Δr_u is the amount of data, or tokens, that is removed from buffer u by allocating the resources to the client u, which is the same as the quantity a_u above. Looking at one resource slot at the time, we would assign the slot to the stream u that would maximize $-\Delta J$ in the current slot, i.e. pick for transmission the stream u that maximizes the value $-g'(r_u)\Delta r_u$ in each slot ($\Delta r_u \leq 0$). Intuitively, this solution picks for transmission the user that yields the largest reduction in the overall cost function (B.5).

$$\Delta J \approx -g'(r_u)\Delta r_u \tag{B.8}$$

Now recall that the quantities $\Delta r_u = -a_u$ and that a_u are constrained by the channel states, representing the amount of data that would be drained from the queues due to a particular scheduling decision. Having a fixed slot size in terms of number of symbols per slot, the amounts Δr_u are directly proportional to the transmission rates η_u in each resource slot. Setting $p\eta_u = -\Delta r_u$, $f_u(r_u) = g'(r_u)$, we obtain (B.1) from (B.8). The summation in (B.1) is only required when multiple queues transmit within the same time slot k, which is built into the feasible rate constraint $\vec{\eta} \in \mathcal{S}_{s[k]}$. Dropping the constant p, that maps the transmission rate to the amount of transmitted data, does not change the discriminatory functionality.

We find that the explicit minimization of (a function of) the queue sizes in (B.5) yields an implicit maximization of the transmission rate that can be explicitly expressed through the linear approximation (B.8). In turn, (B.8) can be translated into (B.1). Using the results from [34, 35], we require for stability that $g'(r_u)$ is a non-decreasing function of r_u, and that $\lim_{r_u \to \infty} g'(r_u) = \infty$.

Appendix C

Words, Symbols, 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.

access point A network component through which a host is accessing the fixed network. In this work, it is a radio station, attached to the fixed network, through which a mobile host is communicating.

access router The first (or last) router that a data packet passes on its way through the network. Hosts communicate directly with access routers on the network layer (although the data packet may pass one or several switches on the link layer).

channel capacity In this thesis it is used as defined in Definition 4.1 on page 58. Our definition is different from the classical Shannon capacity [68].

client A client is a consumer of server resources. A client generates revenue for the owner of the server. In this thesis, the users of the wireless links are regarded as clients in the client-server context, where the radio links are the resources of the server. The schedulers should assign server resources to the clients in a cost-efficient way. In this thesis, the words session, client, host, and user, may be used interchangeably when there is no risk for confusion. However, a user may utilize more than one client.

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. Each layer in the OSI-stack may add its own header to the data packets. To be separated from the content or payload part of a data packet.

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. In this thesis, the words session, client, host, and user, may be used interchangeably when there is no risk for confusion.

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.

mobile host A host that is mobile. In this work, the mobile host is also assumed to be communicating with another host over a radio link, through a (wireless) access point.

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.

router A machine that works as a network node at the OSI-stack 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 clients 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.

session A session is an activity period of a client. During a session, a client may produce work (data packets) that the server (the wireless link) has to serve. The scheduler should assign the server resources cost-efficiently. In this thesis, the words session, client, host, and user, may be used interchangeably when there is no risk for confusion. However, a user may run more than one session.

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

user A user is an owner of a client. In this thesis, the words session, client, host, and user, may be used interchangeably when there is no risk for confusion. However, a user may utilize more than one client.

C.2 Symbols

a_u Allocation vector element, containing the number of BPSK bins assigned to client u by the scheduler.

$B_u(t_0, t_1)$ Achieved transmission during time $t_0 \leq t \leq t_1$.

C_u Average allocated bin capacity, defined in Definition 4.3 on page 59.

$C(u, s)$ Bin capacity, defined in Definition 4.1 on page 58.

$\bar{C}_u$ Average bin capacity, defined in Definition 4.2 on page 58.

d_s Decision vector element, containing the index to a client that has been assigned resource bin s.

e_u Price value from price model for client u.

E_u Cost efficiency, defined in Definition 3.4 on page 49.

h Feedback factor, giving the average number of resource claims per available resource slot, see Example 4.4 on page 66.

H Total amount of resource claim feedback allowed on a link, see Example 4.4 on page 66.

I_u Token rate for client u, admitted by the service level controller (SLC).

$\underline{I}_u$, $\overline{I}_u$ Minimum and maximum rate limits, respectively, assigned by the admission policy (AP).

K, K_D, K_I Gain factors in the PID controller, see Definition 3.3 on page 47.

K_u Individual PID controller gain factor for client u, see Example 3.6 on page 49.

P_u Weighting factor in the quadratic criterion.

P_{ue} Externally assigned priority for the quadratic criterion.

P_{ui} Internally calculated *compensation factor*, to balance the internal priority in the quadratic criterion.

r_u Buffer level, or, token bucket level for client u.

$R_u(t)$ Transmission rate for client u during time slot t. It is defined in Definition 7.1 on page 113.

s Index over resource bins.

S Total number of available resource bins during one scheduling round.

u Index over users, clients, or sessions.

U Total number of active users, clients, or sessions.

C.3 Abbreviations and Acronyms

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

AC Admission Controller. Consists of an AP and an SLC. It performs revenue-aware link load balancing.

AP Admission Policy. The part of the Admission Controller that handles admission and rejection decisions, and sets the service level limits that the SLC must maintain.

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

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.

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.

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

PID Proportional, Integrating, and Differentiating controller. Simple feedback controller with integrating and differentiating action.

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 ($M = 4, R = 2$).

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.

SLC Service Level Controller. Adapts the admitted service levels to the current channel properties, within the limits set by the AP.

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.

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

Bibliography

[1] A. Algoja. A business model for fourth generation wireless mobile networks. In H. Lipmaa and H. Pehu-Lehtonen, editors, *Seminar on Network Security*, Proceedings of the Helsinki University of Technology, 2000. www.tml.hut.fi/Opinnot/Tik-110.501/2000/papers.

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

[3] J. Alm, S. Borgström, and J. Hybinette. Mobila virtuella operatörer - förutsättningar för framgång. Master's thesis, Institute of Economic Research, Lund University, Sweden, 2001.

[4] M. Andrews, K. Kumaran, K. Ramanan, A. Stolyar, R. Vijayakumar, and P. Whiting. CDMA data QoS scheduling on the forward link with variable channel conditions. Technical report, Bell Laboratories, Lucent Technologies, 2000.

[5] M. Andrews, K. Kumaran, K. Ramanan, A. Stolyar, R. Vijayakumar, and P. Whiting. Scheduling in a queueing system with asynchronously varying service rates. *Probability in Engineering and Informational Sciences*, to appear 2004 (previous version as technical memo: CDMA Data QoS Scheduling on the Forward Link with Variable Channel Conditions, Bell Labs, 2000).

[6] M. Andrews, K. Kumaran, K. Ramanan, A. Stolyar, P. Whiting, and R. Vijayakumar. Providing quality of service over a shared wireless link. *IEEE Communications Magazine*, pages 150–154, Feb. 2001.

[7] K. J. Åström and B. Wittenmark. *Computer Controlled Systems - Theory and Design*. Prentice Hall, 3 edition, 1996.

[8] L. Badia, M. Lindström, J. Zander, and M. Zorzi. An economic model for the radio resource management in multimedia wireless systems. *Computer Communications*, 27(11):1056–1064, July 2004.

[9] H. Balakrishnan, E. Amir, S. Seshan, and R. H. Katz. Improving TCP/IP performance over wireless networks. In *ACM MobiCom'95*, pages 2–11, 1995.

[10] G. Barriac and J. Holtzman. Introducing delay sensitivity into the proportional fair algorithm for CDMA downlink scheduling. In *Proc. IEEE Interna-*

tional Symposium on Spread Spectrum Techniques and Applications, volume 3, pages 652–656, 2002.

[11] P. Beming and M. Frodigh. Admission control in frequency hopping GSM systems. In *Proc. IEEE Vehicular Technology Conference*, volume 2, pages 1282–1286, May 1997.

[12] J. C. Bennett and H. Zhang. WF2Q: Worst-case fair weighted fair queueing. In *IEEE INFOCOM Conference on Computer Communications*, volume 1, pages 120–128, 1996.

[13] M. Berlin. Channel modeling and simulation for the mobile internet. Master's thesis, Uppsala University, 2002.

[14] V. Bharghavan, S. Lu, and T. Nandagopal. Fair queueing in wireless networks: Issues and approaches. *IEEE Personal Communications*, pages 44–53, Feb. 1999.

[15] 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.

[16] E. Biyalogorsky and E. Gerstner. Contingent pricing to reduce price risks. Technical report, University of California, Davies, 2002.

[17] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. An architecture for differentiated services. RFC2475, Dec. 1998.

[18] T. Bonald. A score-based opportunistic scheduler for fading radio channels. In *European Wireless Conference*, Barcelona, Feb. 2004.

[19] 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.

[20] C. Bormann, C. Burmeister, M. Degermark, H. Fukushima, H. Hannu, L.-E. Jonsson, R. Hakenberg, T. Koren, K. Le, Z. Liu, A. Martensson, A. Miyazaki, K. Svanbro, T. Wiebke, T. Yoshimura, and H. Zheng. RObust Header Compression (ROHC). RFC3095, July 2001.

[21] R. Braden, D. Clark, and S. Shenker. Integrated services in the internet architecture: an overview. RFC1633, June 1994.

[22] R. Bruno, R. G. Garroppo, and S. Giordano. Estimation of token bucket parameters of VoIP traffic. In *Proceedings of the IEEE Conference on High Performance Switching and Routing*, pages 353–356, June 2000.

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

[24] D. W. Clarke, C. Mohtadi, and P. S. Tuffs. Generalized predictive control - part I. the basic algorithm. *Automatica*, 23(2):137–148, 1987.

[25] D. W. Clarke, C. Mohtadi, and P. S. Tuffs. Generalized predictive control - part II. extensions and interpretations. *Automatica*, 23(2):149–160, 1987.

[26] B. Classon, P. Sartori, V. Nangia, X. Zhuang, and K. Baum. Multi-dimensional adaptation and multi-user scheduling for wireless OFDM systems. In *Proc. IEEE International Conference on Communications*, volume 3, pages 2251–2255, May 2003.

[27] A. Duel-Hallen, S. Hu, and H. Hallen. Long-range prediction of fading signals. *SigMag*, 17(3):62–75, May 2000.

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

[29] T. Ekman. *Prediction of Mobile Radio Channels - Modeling and Design*. Doctoral Thesis, Uppsala University, Oct. 2002.

[30] T. Ekman, M. Sternad, and A. Ahlén. Unbiased power prediction on broadband channels. In *Proc. IEEE Vehicular Technology Conference*, Vancouver, Canada, Sept. 2002.

[31] V. Erceg, L. J. Greenstein, S. Y. Tjandra, S. R. Parkoff, A. Gupta, B. Kulic, A. A. Julius, and R. Bianchi. An empirically based path loss model for wireless channels in suburban environments. *IEEE Journal on Selected Areas in Communications*, 17(7):1205–1211, July 1999.

[32] N. C. Ericsson. *On Scheduling and Adaptive Modulation in Wireless Communications*. Licentiate Thesis, June 2001.

[33] 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.

[34] A. Eryilmaz, R. Srikant, and J. R. Perkins. Stable scheduling policies for fading wireless channels. Technical report, University of Illinois, Urbana-Champaign, http://www.comm.csl.uiuc.edu/˜ srikant, 2002.

[35] A. Eryilmaz, R. Srikant, and J. R. Perkins. Stable scheduling policies for fading wireless channels. In *Proc. IEEE International Symposium on Information Theory*, page 454, Yokohama, Japan, June 2003.

[36] S. Falahati. Convolutional codes for wireless packet data systems. Licentiate Thesis, Chalmers University of Technology, Feb. 2000.

[37] S. Falahati. *Adaptive Modulation and Coding in Wireless Communications with Feedback*. PhD thesis, Chalmers University of Technology, Göteborg, Sweden, Oct. 2002.

[38] S. Falahati, N. C. Ericsson, and A. Ahlén. Overhead tradeoffs in wireless cellular networks - optimal design guide.

[39] 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.

[40] 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.

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

[42] D. Gesbert and M.-S. Alouini. Selective multi-user diversity. In *Proc. IEEE International Symposium on Signal Processing and Information Technology*, 2003.

[43] M. Handley, S. Floyd, J. Padhye, and J. Widmer. TCP friendly rate control (TFRC). RFC3448, Jan. 2003.

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

[45] J. M. Holtzman. Asymptotic analysis of proportional fair algorithm. In *Proc. IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, pages F33–F37, 2001.

[46] 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.

[47] M. Johansson. Diversity-enhanced equal access - considerable througput gains with 1-bit feedback. In *IEEE SPAWC, Workshop on Signal Processing in Wireless Communications*, Lisbon, Portugal, July 2004.

[48] M. Johansson. *Resource Allocation under Uncertainty - Applications in Wireless Communications*. PhD thesis, Uppsala Universtiy, 2004.

[49] 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.

[50] M. Kazmi, P. Godlewski, and C. Cordier. Admission control strategy and scheduling algorithms for downlink packet transmission in WCDMA. In *Proc. IEEE Vehicular Technology Conference*, pages 674–680, Sept. 2000.

[51] K. Kim, H. Kim, and Y. Han. A proportionally fair scheduling algorithm with QoS and priority in 1XEV-DO. In *Proc. IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, pages 2239–2243, 2002.

[52] R. Koodli. Fast handover for mobile IPv6. INTERNET DRAFT, Jan. 2004. MIPSHOP Workgroup, IETF.

[53] 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.

[54] 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.

[55] S. Netessine and R. Shumsky. Introduction to the theory and practice of yield management. *INFORMS Transactions on Education*, 3(1):34–44, September 2002.

[56] K. Norlund, T. Ottosson, and A. Brunström. Fairness measures for best effort traffic in wireless networks. In *Proc. IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, Barcelona, Spain, Sept. 2004.

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

[58] J. Öhr. *Anti-windup and Control of Systems with Multiple Input Saturations*. PhD thesis, Uppsala University, 2003.

[59] P. Olla and N. V. Patel. A value chain model for mobile data service providers. *Telecommunications Policy*, 26(9-10):551–571, October-November 2002.

[60] G. W. Ozanich, C.-W. Hsu, and H. W. Park. 3-G wireless auctions as an economic barrier to entry: the Western European experience. *Telematics and Informatics*, 21(3):225–234, Aug. 2004.

[61] A. K. Parekh and R. G. Gallager. A generalized processor sharing approach to flow control in integrated services networks: The single-node case. *IEEE/ACM Transactions on Networking*, 1(3):344–357, June 1993.

[62] D. Park, H. Seo, H. Kwon, and B. G. Lee. A new wireless packet scheduling algorithm based on the cdf of user transmission rates. In *Proc. IEEE Global Telecommunications Conference*, San Francisco, Dec. 2003.

[63] J. Postel. User datagram protocol. RFC768, Aug. 1980.

[64] J. Postel. Transmission control protocol. RFC793, 1981.

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

[66] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RTP: A transport protocol for real-time applictions. RFC1889, Jan. 1996.

[67] S. Shakkottai and A. L. Stolyar. Scheduling algorithms for a mixture of real-time and non-real-time data in HDR, Sept. 2001.

[68] C. E. Shannon. A mathematical theory of communication. *The Bell System Technology Journal*, 27, 1948.

[69] A. Silberschatz and P. B. Galvin. *Operating Systems Concepts*. Addison-Wesley Publishing Company, Inc., 4th edition, 1994.

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

[71] Statens Järnvägar. Historien om SJ, 2001.

[72] M. Sternad. *Modelling and Control - Lecture Notes for the Project Oriented Course in Process Control*. Uppsala University, Department of Technology, 1994.

[73] M. Sternad and D. Aronsson. Channel estimation and prediction for adaptive OFDM downlinks. In *Proc. IEEE Vehicular Technology Conference*, Orlando, FL, Oct. 2003.

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

[75] M. Sternad, T. Ottosson, A. Ahlén, and A. Svensson. Attaining both coverage and high spectral efficiency with adaptive OFDM downlinks. In *Proc. IEEE Vehicular Technology Conference*, Orlando, FL, Oct. 2003.

[76] A. L. Stoylar and K. Ramanan. Largest weighted delay first scheduling: Large deviations and optimality. *The Annals of Applied Probability*, 11(1):1–48, 2001.

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

[78] L. Tassiulas and A. Ephremides. Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks. *IEEE Transactions on Automatic Control*, 37(12):1936–1948, Dec. 1992.

[79] L. Tassiulas and A. Ephremides. Dynamic server allocation to parallel queues with randomly varying connectivity. *IEEE Transactions on Information Theory*, 39(2):466–478, Mar. 1993.

[80] Technical Specification Group - Services and System Aspects. QoS concept and architecture. Technical Report 3G TS 23.107 v5.11.0, 3rd Generation Partnership Project (3GPP), 2003.

[81] Technical Specification Group - Services and System Aspects. Services and service capabilities. Technical Report 3G TS 22.105 v6.2.0, 3rd Generation Partnership Project (3GPP), 2003.

[82] 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.

[83] 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.

[84] Vodafone. http://3g.vodafone.se, July 2004.

[85] M. Welzl. Scalability and Quality of Service: A Trade-off? *IEEE Communications Magazine*, pages 32–36, June 2003.

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