The Dissertation Committee for Shailesh Patil
certifies that this is the approved version of the following dissertation:

# Opportunistic Scheduling and Resource Allocation Among Heterogeneous Users in Wireless Networks

Committee:

Gustavo de Veciana, Supervisor

Vijay K. Garg

Harrick Vin

Sanjay Shakkottai

Sriram Vishwanath

# Opportunistic Scheduling and Resource Allocation Among Heterogeneous Users in Wireless Networks

**by**

**Shailesh Patil, B.S., M.S.E**

**DISSERTATION**

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

**DOCTOR OF PHILOSOPHY**

THE UNIVERSITY OF TEXAS AT AUSTIN

May 2006

Dedicated to my parents, who have always encouraged me.

# Acknowledgments

I want to thank my advisor Prof. Gustavo de Veciana for being the great advisor he has been. The longer I work with him, the more I realize that it is impossible to find a better advisor. I also want to thank my committee members Prof. Harrick Vin, Prof. Vijay Garg, Prof. Sanjay Shakkottai and Prof. Sriram Vishwanath for their insightful comments on my work.

I am eternally indebted to my parents. My late father always guided me and supported me in all my endeavors. I want to thank my mother for facing the hardship of living in a foreign land so that I could pursue graduate studies. My brother, sister and brother-in-law have always encouraged me to pursue the career of my liking, and I am grateful to them for that.

I would also like to thank my friend Aamir Hasan with whom I have had countless discussion about why Indian cricket team is better than that of Pakistan's. I also want to thank my friends Alejandro Icaza and Bilal Sadiq for making my stay at UT an enjoyable one.

SHAILESH PATIL

May 2006

# Opportunistic Scheduling and Resource Allocation Among Heterogeneous Users in Wireless Networks

Publication No. _____

Shailesh Patil, Ph.D.
The University of Texas at Austin, 2006

Supervisor: Gustavo de Veciana

This dissertation studies and proposes new methods to perform opportunistic scheduling in different scenarios for centralized wireless networks. We first study the performance of measurement-based opportunistic scheduling strategies in practical scenarios where users' heterogenous capacity distributions are unknown. We make the case for using maximum quantile scheduling, i.e., scheduling a user whose current rate is in the highest quantile relative to its current empirical distribution. Under fast fading, we prove a bound on the relative penalty associated with such empirical estimates, showing that the number of independent samples need only grow linearly with the number of active users. Furthermore, we show several desirable properties of maximum quantile scheduling for the saturated regime, and give bounds on performance under the dynamic regime.

Next we propose a novel class of opportunistic scheduling disciplines to handle mixes of real-time and best effort traffic at a base station. The objective is to support probabilistic service rate guarantees to real-time sessions while still achieving opportunistic throughput gains across users and traffic types. Under fast fading and maximum quantile scheduling, we are able to show a stochastic lower bound on the service a real-time session would receive. Such

bounds are critical to enabling predictable quality of service and thus the development of complementary resource management and admission control strategies. Our simulation results show that the scheme can achieve more than 90% of the maximum system throughput capacity while satisfying the QoS requirements for real-time traffic.

Finally, we propose methods to reduce the feedback overhead for users' channel state information needed for opportunistic scheduling at a base station. We first propose a contention based scheme known as 'static splitting' for a best effort traffic only scenario. Next we consider reducing feedback overhead in a system supporting a mixture of best effort and real-time traffic. We argue that one needs to combine contention based schemes with polling subsets of users to reduce the amount of feedback needed to exploit opportunism, and yet meet real-time users' QoS guarantees. Based on this argument we propose a joint polling and opportunistic scheduling (JPOS) scheme.

# Table of Contents

# List of Tables

# List of Figures

# Chapter 1

# Introduction

## 1.1 Motivation

Wireless communication devices like cellular phones have become ubiquitous, fulfilling to a large extent the promise of 'anywhere-anytime' communication. While these devices are conventionally used for voice, they are now increasingly being used to provide data services. New standards like WiMAX and the family of IEEE 802.11 standards have/are being devised to make wireless broadband access possible. These technologies promise to give users high speed access to the Internet along with various other data services such as video, images and text messaging.

However, in order to provide this multitude of services, several issues need to be addressed. One major issue is 'efficient' scheduling and allocation of the finite resources (e.g., bandwidth) available at the wireless service provider among users. In this dissertation we study this issue and propose solutions to address it.

Scheduling and resource allocation can be a fairly challenging problem even in traditional wireline systems due to differing requirements of users. However, a key feature of wireless systems relative to wireline systems is that, the channel capacity, or service rate seen by users, may exhibit temporal variations. This uncertainty in channel capacity or rate variation poses several additional challenges to efficient scheduling and resource allocation. Specifically:

- Users near a base station will generally experience a much better channel capacity than users at the edge of the cell.

- Each user may undergo short term fading which is location dependent.
- The statistics of such short term variations may not be identical across users.
- Short term fading statistics are also time varying, this is especially true for mobile users.
- A scheduling scheme should ideally not only be able to handle the uncertainty, but also be able to exploit it, i.e., opportunistically serve users with good channels.
- A good scheduling scheme should enable viable complementary resource allocation and connection admission control strategies that take into account the time varying nature of the wireless channel.

In summary, not only are the rates seen by users are time varying, but the variations ***heterogeneous*** across users. Good scheduling and resource allocation solutions should not only be able overcome the heterogeneity and the variability, but also ***opportunistically*** exploit it.

## 1.2 Related Work

There has been a substantial amount of research in this area, with some progress towards addressing the above challenges. Knopp et. al. [16] were the first to propose a scheduling discipline *max rate* that opportunistically exploited the uncertainty in channel capacities among users generating best effort data traffic. However, their scheme suffered from being unfair to users with lower channel capacities. *Proportionally fair* [13] and *exponential rule* [33] scheduling were introduced to address this issue. These disciplines have also been shown to potentially give quality of service (QoS) guarantees to users generating real-time traffic in [32]. Wu. et. al. proposed using the concept of 'effective bandwidth' to provide QoS guarantees to users. Opportunistic

scheduling requires a fairly large amount of feedback at the scheduler, so several researchers have studied ways to reduce the feedback in [27][35][11]. These and other related work will be discussed in more detail in later chapters. Yet, for now we will note that several challenges in dealing with heterogeneity in channel capacities among users and devising concrete resource allocation and call admission strategies remain.

## 1.3 Organization of the Thesis

We describe the organization of this thesis now. As discussed above, both the demands and the rate or channel capacities realized by users can be heterogeneous. This is usually addressed in the literature by attaching different priority weights to users. These weights are mostly determined using measurement, and it is important to understand the effect of measurement based errors. In Chapter 2, we study this and make the case for a scheduling discipline that we call *maximum quantile scheduling*. We also study several properties of maximum quantile scheduling under different scenarios.

In Chapter 3 we study the case where the offered load includes a mixture of real-time streams and best effort data sessions. An opportunistic scheduling and resource allocation scheme is developed and is combined with maximum quantile scheduling to give QoS guarantees to real-time streams. The proposed scheduling scheme exploits opportunism across all users, and a concrete resource allocation strategy allows the development of a complementary call admission strategy for real-time streams.

Opportunistic scheduling requires that every time a resource allocation decision be made, the scheduler should know the current channel capacity of all users. This requires all users to feed back their current channel capacity for every opportunistic decision being made. In Chapter 4 we propose a scheme that significantly reduces the amount of feedback required while realizing most of the potential gains of opportunism. Chapter 5 concludes this thesis.

# Chapter 2

# Measurement-Based Opportunistic Scheduling

## 2.1 Introduction

***Motivation.*** As discussed in the introduction, the channel capacities seen by users are not only time varying, but are also heterogeneous, e.g., users close to a base station see significantly different channel capacity than those further off. Thus it is important to devise opportunistic schedulers that do not starve some users, e.g., those with poor channels, to achieve some degree of fairness among users sharing a base station. To this end many opportunistic scheduling schemes have been devised that make decisions by selecting the user that currently has the highest weighted channel capacity. In practice the weights may be hard to determine, because they depend in a complex way on the users' channel capacity distributions, the number of users, and the characteristics of their traffic. Thus they either need to be estimated or tuned based on the service users have received or their queue lengths.

Unfortunately, the complex dependence of weights may make them very sensitive to changes in the system, i.e., if a user's traffic characteristics change, or a user leaves or enters the system, or the channel characteristics of a user change (e.g., a mobile user comes out of the shadow of a building), then the weights associated with *all* users may need to change. Therefore, it is likely that a significant fraction of time will be spent estimating/tuning weights to their 'ideal' values. In fact, if the system is dynamic enough and/or the tuning algorithm is not sensitive enough, one may never converge, possibly compromising fairness but also, and more importantly leading to poor throughput performance. Consider a simple example. Due to the stochastic or time vary-

ing nature of channel capacity and users' traffic, a measurement-based opportunistic scheduler may be biased in favor of a user who has not received service in the recent past or one that currently has a high queue. While this myopic approach is good for short term fairness, the scheduler may end up serving a user even though it is not currently experiencing a good channel rate. This in turn decreases the achieved opportunism and long term throughput the system can sustain. In heavily loaded systems, at a given moment of time, it is very likely that there exists a group of users which are starved. If those users are served, others may become starved, leading to a cycle, in which the level of opportunism and throughput are low. In this chapter we will see that indeed the performance of many proposed opportunistic scheduling schemes in such regimes are subject to such performance penalties.

Recently, distribution based opportunistic schedulers have been proposed by several researchers under different guises [22][23][4][27]. We shall refer to this family of schemes as *maximum quantile schedulers*. The idea is to schedule a user whose current rate is highest relative to his *own* distribution, i.e., in the highest quantile. As will be explained in the sequel because the quantile of each user's rate is uniformly distributed, maximum quantile scheduling is automatically temporally fair – i.e., no weights required to achieve fairness. However, in practice maximum quantile scheduling would involve estimating each user's channel capacity distribution. In this chapter we will show that the throughput penalty incurred from estimating user's distributions can be limited. Furthermore, unlike other schemes, maximum quantile does not require estimation/tuning of weights which depend on users' joint channel capacity distributions, and so it is robust to fast changes in the number of users and/or their activity levels. In other words the performance penalties associated with estimation/tuning are substantially reduced.

***Contributions.*** The following is the list of the key contributions of this chapter:

- We investigate the throughput performance of maximum quantile scheduling and show that if the achievable instantaneous rate of users' is bounded, then among the class of scheduling policies that serve each user an equal fraction of time, maximum quantile scheduling maximizes the long term system throughput when there is a large number of users. Furthermore, we show that the marginal distribution for the rate when users are selected for service under maximum quantile scheduling can not be stochastically dominated by any other non-idling scheduler.

- Under the assumption of fast fading, we prove a bound on each user's relative throughput penalty when maximum quantile scheduling is based on empirical distributions for users' channel capacity. This is significant because it shows that such penalties can be controlled if the number of independent samples used to estimate the empirical distribution is roughly proportional to the *number of users in the system*. Thus maximum quantile scheduling can be used even when users' channel distributions are not known or slowly changing.

- We conjecture that the best way to serve a user (especially under heavy loads) is to serve it when its rate is high compared to its distribution, rather than favoring a user that has not been served for some period of time. This conjecture is supported by simulating the performance of various measurement based opportunistic scheduling schemes for various network and traffic scenarios. We find that maximum quantile scheduling can have significantly better performance in terms of both packet delay and file transfer delay, e.g., up to 40% improvement. We stress that this observation has not been made by previous work.

- Finally, we study a dynamic saturated system (i.e., one where the number of users are changing) served under maximum quantile scheduling. We show simple upper and lower bounds on transfer delays based on

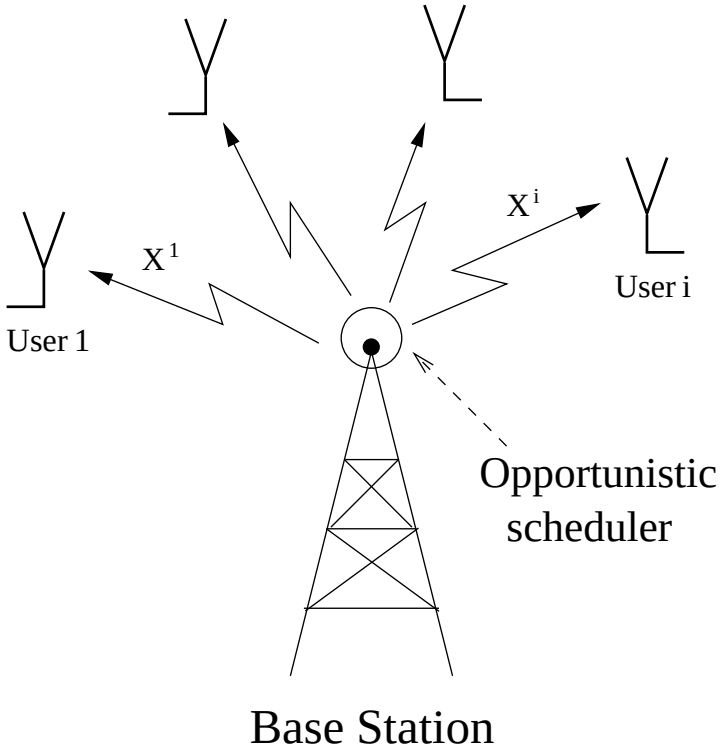



Figure 2.1: Downlink scheduling to users from a base station.

multiclass M/GI/1 processor sharing queues and a parametric model for channel variations.

***Chapter Organization.*** This chapter is organized as follows. In Section 2.2 we revisit and critique representative prior work in the area of opportunistic scheduling and introduce some known features of maximum quantile scheduling. Throughput performance and optimality of maximum quantile scheduling is studied in Section 2.3. We prove our bound on the relative throughput penalty associated with measuring distributions in Section 2.4. Simulation results comparing the performance of maximum quantile scheduling to other schemes are presented in Section 2.5 and 2.6. Section 2.7 concludes this chapter.

## 2.2 Revisiting Opportunistic Scheduling

### 2.2.1 System Model and Notation

We begin by introducing our system model and some notation. For simplicity, we focus on downlink scheduling from a base station to multiple users (see Figure 2.1). Suppose time is divided into equal sized slots and at most one user gets served per slot, e.g., for the CDMA-HDR systems defined in the CDMA2000 IS-856 standard, the slot time has a duration of 1.67 ms [2]. During each slot, each user feeds back the data rate it can support and the base station makes a decision on which user should get served. In the sequel we use the terms 'channel capacity' and 'rate' interchangeably and make the following assumption on user's channel characteristics over time slots.

***Channel assumptions.*** In practice, characterizing the channel capacity or rate seen by a user is quite complicated. There are several factors that affect the capacity, they can be broadly classified into two classes [28][12]. First, there is large scale path loss in the 'average' capacity seen by a user due to the distance of a user from the base station, and the shadowing due to obstacles in the path between the user and the base station. Secondly, there is small-scale variation or fading in the instantaneous capacity due to multipath time delay spread, the speed of such variations depends on the Doppler spread seen by the user. Therefore a simple yet reasonably accurate model may be to view the channel capacity seen by a user as a quasi stationary random process, with the marginal distribution that changes following changes in the large scale path loss. These marginal distributions are likely to be different across users.

In this chapter we will assume such quasi-stationary characteristics for users' channel capacities, and for analysis purposes assume the regime where the users' channels are in fact stationary. The following is a formal statement of our assumptions on the channel capacity distributions(s) across slots.

**Assumption 2.2.1.** *We assume the channel capacity (rate) for each user is a stationary ergodic process and these processes are independent across users. Further we assume that the marginal distribution for each user is continuous and is either known a priori, or estimated by the base station.*

***Discussion on the channel assumption.*** We conjecture that the channel should remain stationary for roughly $O(n^2)$ (here $n$ is the number of users in the system) samples for the result on measurement based performance proved in Section 2.4 to be practically viable. The assumption that users' rates are independent is also likely to be true, though a notable exception is the case where mobile users move in a correlated manner, e.g., along a highway. The assumption that the base station knows, and in particular can estimate, the marginal distributions of the channel capacity processes may seem unreasonable, but simple book keeping on the users' feedback of the currently achievable rate can be used to estimate distributions. We will discuss estimation of such distributions in Section 2.4. Note that channel capacities are not restricted to any specific distribution, or class of distributions, i.e., users can undergo any fading process. This makes the analysis presented later applicable to real world scenarios. Note that the we require the marginal distributions of rates to be continuous only for simplicity sake, we will indicate how the results presented here can be extended to the discrete case.

***Notation.*** In the sequel we will let $x^i(t)$ denote the realization of the channel capacity of User $i$ at time slot $t$, and let $X^i$ be a random variable whose distribution is that of the channel capacity of User $i$ on a *typical* slot. Recall that we will be assuming $X^i$ to be independent across users but need not be identically distributed. We denote the distribution function of $X^i$ by $F_{X^i}(\cdot)$. For simplicity, we will assume that $F_{X^i}(\cdot)$ is a strictly increasing function, so that its inverse denoted by $F^{-1}_{X^i}(\cdot)$ is defined.

***System Scenario.*** There are several system scenarios (Figure 2.2) one can consider. In a real world scenario, the number of users in the system

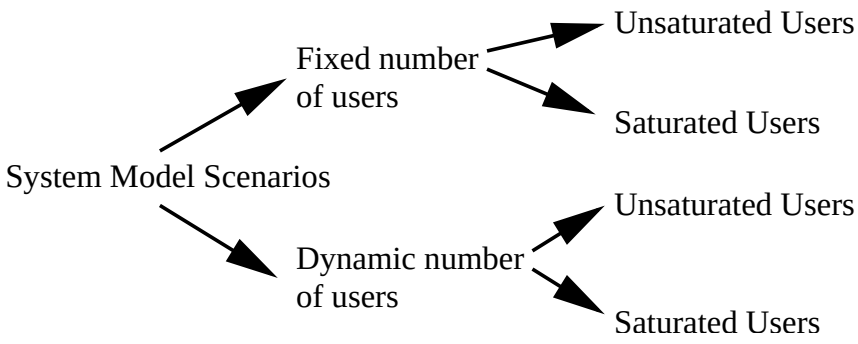



Figure 2.2: Different scenarios for system model.

may be changing, and users may not be infinitely backlogged, i.e., unsaturated dynamic. However, such a scenario is analytically intractable, therefore we usually study different idealizations. The first idealization is the 'fixed saturated' case, where the number of users in the system does not change with time and each user is infinitely backlogged. Such a scenario is an approximation where the number of users in the system changes slowly and packet queues for each user are always non empty at the base station. This idealization is often studied in the literature, and we will largely focus on this case. We will also study the 'fixed unsaturated' and 'dynamic saturated' cases, the former referring to the scenario where even though the number of users remain static, they are not necessarily backlogged, and the later refers to the scenario where the number of users changes with time, but whenever a user is present, it is infinitely backlogged.

We denote the number of users present in the system on slot $t$ by $n(t)$. We simplify this to $n$ in a fixed system (saturated or unsaturated) since the number of users is constant. The set of active, i.e., backlogged users on slot $t$ is denoted by $A(t)$. In other words, $A(t)$ is the set of users that wish to be served on slot $t$. Note that in a dynamic saturated system $|A(t)| = n(t)$, while in fixed saturated system $|A(t)| = n$.

### 2.2.2 Weight based Opportunistic Scheduling Schemes

Opportunistic scheduling was first proposed in [16]. They proposed *maximum rate scheduling*, where the user with maximum channel capacity at that point of time is served, i.e., User $k(t)$ is selected for service on time slot $t$ if

$$k(t) \in \arg\max_{i \in A(t)} x^i(t).$$

This maximizes system throughput in a fixed saturated system, but in a system where users have heterogenous rate distributions, may neglect those with poor channels.

Subsequently a myriad of approaches have been proposed to address both unfairness and/or performance issues. One of the more cited schemes is *proportional fair scheduling* [13][37][7] which serves the user whose current rate normalized by a moving average of his allocated rate is the highest, i.e., User $k(t)$ is selected for service at time slot $t$ if

$$k(t) \in \arg\max_{i \in A(t)} \frac{x^i(t)}{\mu^i(t)}, \tag{2.1}$$

where

$$\mu^i(t) = (1 - \frac{1}{t_c})\mu^i(t-1) + \frac{1}{t_c} x^i(t) \mathbf{1}_{S^i_{pf}(t)}$$

and $t_c$ is the moving average parameter, $S^i_{pf}(t)$ is the event that User $i$ gets served on slot $t$ by the scheme, and $\mathbf{1}_{S^i_{pf}(t)}$ is the indicator function of $S^i_{pf}(t)$.

As a simple experiment we compare the throughput achieved by proportionally fair to that achieved by maximum quantile scheduling (described in the next subsection) in a fixed saturated system. Our setup consists of two classes of users having a mean signal to noise ratio (SNR) of 2 and 0.1, with both classes experiencing Rayleigh fading and containing an equal number of users. The Doppler frequency for the channel capacity variation of all users is set to 15Hz, and the slot size is set to 1.67 msec. The bandwidth associated with each user is 500 KHz and we assume that coding achieves the Shannon rate [9]. This setup will be used throughout the chapter for simulations, and

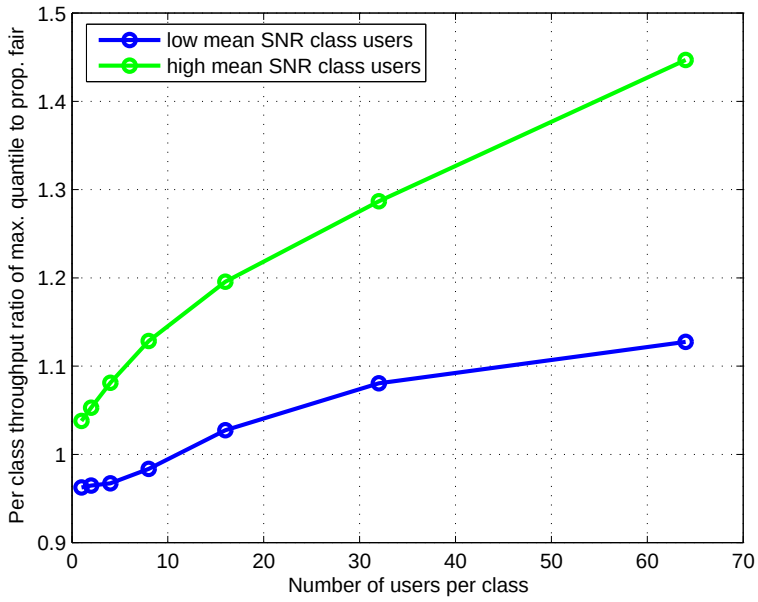



Figure 2.3: Ratio of per class throughput achieved by maximum quantile scheduling to that achieved by proportional fair.

unless specified otherwise, both classes will contain an equal number of users. The parameter $t_c$ is set equal to 1000 slot lengths [13].

Figure 2.3 exhibits the ratio of the per class throughput achieved by maximum quantile scheduling versus that achieved by proportional fair based on allocated rate for an increasing number of users. As can be observed, maximum quantile achieves significant gains in throughput (up to 45%) for users having lower mean SNR, while one observes more than 10% gain for users having high mean SNR. This clearly illustrates that scheduling based on the recent service given to a user can lead to loss in opportunism.

Queue based opportunistic scheduling schemes that factor the magnitude of ongoing users' queue lengths (as a measure of recent service given to a user) and their channel capacity in deciding which to serve have also been proposed in the literature. For example, the *exponential rule* [33][32][31], chooses

to serve User $k(t)$ on slot $t$ if

$$k(t) \in \arg\max_{i \in A(t)} [\gamma^i x^i(t) \exp(\frac{a^i q^i(t)}{1 + \sqrt{\bar{q}(t)}})],$$

where $q^i(t)$ is the queue length of User $i$ at time $t$, $a^i$ is the weight associated with User $i$'s queue, $\bar{q}(t)$ is the average weighted queue length across users at time $t$, and $\gamma^i$ is the weight associated with User $i$'s channel rate $x^i(t)$. Factoring the users' queue length has the potential advantage of reducing packet delays. Indeed, it has been shown in [31] that under a heavy traffic scenario, the exponential rule will minimize the maximum of weighted queue length, i.e., $a^i q^i$. Good packet delay performance of the rule has been supported by simulations shown in [32]. We will revisit this point in Section 2.5, and show that in practice, not unlike proportional fair, such queue based schemes introduce biases that may compromise opportunism thus compromising packet delay performance.

Finally, [18][19][17] proposed strategies that maximize system throughput under fairness constraints. For example, they show that a scheduling policy of the form

$$k(t) \in \arg\max_{i \in A(t)} [x^i(t) + \nu^i], \tag{2.2}$$

maximizes the overall sum/system throughput subject to constraints on the fraction of time each User $i$ is served in a fixed saturated regime. Here $\nu^i$ is a weight associated with User $i$ that ensures that users get served the desired fraction of time. Similar optimal schemes were proposed for rate and utility based fairness.

While the fairness and optimality characteristics of these schemes are desirable, in practice they would require estimating thresholds $\nu^i$ which are complicated functions of users' rate distributions, number of users and temporal constraints. In the sequel (Section 2.4 and 2.6), we show that such estimates may converge slowly and are not robust to changes in the unsaturated and/or dynamic regimes.

### 2.2.3 Maximum Quantile Scheduling

Maximum quantile scheduling has been proposed independently by several researchers. Specifically [22][23] proposed a 'CDF based scheme'. While [4] proposed a so called 'score based scheduler' and [27] proposed a 'distribution fairness' based scheduler. A variation of the score based scheduler is proposed in [38].

Let us briefly introduce this scheme in the fixed saturated regime. The main idea is to schedule a user whose current rate is highest compared to his *own* distribution, i.e., serve User $k(t)$ during slot $t$ if

$$k(t) \in \arg \max_{i=1,\ldots,n} F_{X^i}(x^i(t)). \tag{2.3}$$

It is well known that $F_{X^i}(X^i)$ is uniformly distributed on $[0,1]$. Let $U^i = F_{X^i}(X^i)$, then $U^i$ is also uniformly distributed on $[0,1]$. Maximum quantile can be thought of as picking the maximum among independent realizations of users' (i.i.d.) $U^i$'s on every slot. Thus, it is clear that maximum quantile is equally likely to serve any user on a typical slot, and as a result all users get served an equal fraction of time, i.e., $\frac{1}{n}^{th}$ of time.

Let $U^{(n)} = \max[U_1, \ldots, U_n]$, where $U_j$ is uniformly distributed on $[0,1]$ $\forall j = 1, \ldots, n$, then

$$\Pr(U^{(n)} \leq u) = u^n, \ \forall u \in [0,1]. \tag{2.4}$$

Then the rate distribution seen by User $i$ on a slot that it gets served is the same as $F^{-1}_{X^i}(U^{(n)})$. Therefore, the average throughput seen by User $i$ is given by $G^i_{mq}(n)$ [22][23],

$$G^i_{mq}(n) = \frac{E[F^{-1}_{X^i}(U^{(n)})]}{n} = \frac{E[X^{i,(n)}]}{n}.$$

where $X^{i,(n)}$ is maximum of $n$ i.i.d. copies of $X^i$, i.e., $X^{i,(n)} := \max[X^i_1, \ldots, X^i_n]$, where $X^i_j \sim X^i$, $\forall j = 1, \ldots, n$. Note that by contrast with the schemes discussed in the previous subsection, if the users' rate distributions were known,

it is fairly easy to evaluate the individual and system throughput for maximum quantile scheduling.

Maximum quantile scheduling can be modified to serve users different fractions of time using easily tuned (distribution independent) weights, see [22][23] for details.

We now digress a bit to discuss maximum quantile scheduling when rates supported by users' are discrete, i.e., $X^i$ is a discrete random variable. For simplicity we will assume that all users share the same support set of rates denoted by $x_j, j = 1, \ldots, l$, and let us denote the probability mass function of User $i$ on these rates by $p^i(\cdot)$. The unique inverse of the cumulative distribution function $F_{X^i}(\cdot)$ is given by

$$F^{-1}_{X^i}(\alpha) = \min_{z \geq 0}\{z \mid F_{X^i}(z) \geq \alpha\}.$$

The discrete case is more subtle because $F_{X^i}(X^i)$ is no longer uniformly distributed on $[0, 1]$. This can be remedied by introducing randomization into the scheduling policy in such a way that the throughput and fairness properties of maximum quantile scheduling under continuous remain preserved. We define maximum quantile scheduling for such users as selecting User $k(t)$ on time slot $t$ if

$$k(t) \in \arg\max_{i=1,\ldots,n} [F_{X^i}(x^i(t)) - p^i(x^i(t))W^i(t)], \tag{2.5}$$

where $w^i(t)$ are realizations of $W^i(t)$, i.i.d. uniformly distributed random variables across users and slots. With this setup one can get the desired property of selecting the maximum among i.i.d. random variables uniformly distributed on $[0, 1]$. The following lemma proves our claim, with the proof given in Appendix 2.8.1.

**Lemma 2.2.2.** *Suppose $X^i$ have discrete distributions, and let*

$$U^i = F_{X^i}(X^i) - p^i(X^i)W^i, \ i = 1, \ldots n$$

*where $W^i$ are uniformly distributed on $[0,1]$ then $U^i$ are uniformly distributed on $[0,1]$ and under Assumption 2.2.1 $U^i$ are independent of each other.*

We note that even though maximum quantile scheduling can be used both in the discrete and continuous rate cases. However it is unlikely that the users' cumulative distribution functions are known at the scheduler, therefore the implementation of the scheduling scheme will be different from above. We will describe the implementation in Section 2.4.

Overall, it is clear that maximum quantile scheduling has some very desirable properties: e.g., it is temporally fair, it is amenable to performance prediction in the fixed saturated case, and Figure 2.3 indicates that it has good throughput performance. However, as discussed earlier, it is unlikely that rate distributions will be known. It is unclear how maximum quantile's performance compares to that of other schemes when distributions are estimated, especially in scenarios other than fixed saturated. In the sequel we will address these issues.

## 2.3 Performance of Maximum Quantile Scheduling in Fixed Saturated System

In this section, we look at two metrics to study the performance of maximum quantile scheduling : (1) the amount of opportunism exploited by the scheme, (2) the throughput achieved by the scheme.

***'Opportunistically' Optimal.*** Suppose we consider as measure of opportunism achieved by User $i$ as the quantile of the rate achieved by the user, i.e., $F_{X^i}(x^i(t))$ whenever it is served. A high quantile means a high degree of opportunism and $E[\sum_{i=1}^{n} F_{X^i}(X^i)\mathbf{1}_{S^i_\beta}]$ denotes the overall expected opportunism realized by a scheduling scheme $\beta$. (Here $S^i_\beta$ is the event that User $i$ is selected for service on typical slot by $\beta$.) It should be clear that maximum quantile scheduling maximizes the system opportunism, and does so while serving all users an equal fraction of time.

***Not Stochastically Dominated.*** Maximum quantile scheduling has an optimality in terms of the rates seen by users in the typical slots where they are served. Let us first introduce the concept of stochastic dominance, before presenting the bound. We say that a random variable $Y$ stochastically dominates random variable $V$, if $\forall v$, $\Pr(Y > v) \geq \Pr(V > v)$. This is denoted as $Y \geq^{st} V$ and it follows that for any increasing function $g(\cdot)$, we have that $g(Y) \geq^{st} g(V)$.

Let $R^i_{mq}$ represent the rate distribution seen by User $i$ when selected for service on a typical slot by maximum quantile scheduling, and let $\overrightarrow{R}_{mq} = (R^1_{mq}, \ldots, R^n_{mq})$, i.e., the vector of random variables representing the rate distributions. Let $\overrightarrow{R}_\beta = (R^1_\beta, \ldots, R^n_\beta)$ be the same quantity for another distinct non idling scheduling scheme $\beta$ that may *not* serve all users an equal fraction of time. By distinct we mean that the scheme does not always pick the user with the maximum quantile, and by non idling, we mean that the scheme will never be idle as long as there is at least one backlogged user. Then our claim is that $\overrightarrow{R}_\beta \not\geq^{st} \overrightarrow{R}_{mq}$, i.e., $\exists j$, such that $R^j_\beta \not\geq^{st} R^j_{mq}$. This is formally stated in the following theorem with the proof given in Appendix 2.8.2.

**Theorem 2.3.1.** *Consider a fixed saturated system with n users, whose channel capacity variations satisfy Assumption 2.2.1. Then for any distinct non idling scheme $\beta$,*

$$\overrightarrow{R}_\beta \not\geq^{st} \overrightarrow{R}_{mq}.$$

Note that a scheduling scheme $\gamma$ is known to be Pareto optimal if there exists no other scheduling scheme that is able to give an equal or higher average throughput to *all* the users than that received by users under $\gamma$. Theorem 2.3.1 can be thought to be a weak form of Pareto optimality in terms of rate seen in a typical slot, not average throughput. We will next show that maximum quantile is not Pareto optimal in terms of average throughput.

***Not Pareto Optimal.*** We illustrate this with a simple pathological example where users' support only discrete rates. (The example can be extended to the continuous case.) Consider a two user system with ON-OFF channels. The ON and OFF channel states correspond to rates 1 and 0 respectively. Users 1 and 2 have an ON probability of 0.6 and 0.4 respectively. Here maximum quantile will serve User 1 at a rate of 0.42, and User 2 at a rate of 0.32. However, it can be shown that maximum quantile may sometimes serve User 2 in OFF state, even though User 1's channel is ON. Therefore, it is possible to improve performance while still serving each user an equal fraction of time. Consider a scheme that always serves the user with the highest instantaneous rate and breaks ties $\frac{7}{24}^{th}$ of times in favor of User 1. Such a scheme will give User 1 a rate of 0.43, and User 2 will get a rate of 0.33. Hence one can give better performance to both the users, while maintaining temporal fairness.

***Throughput Optimal for Large Number of Users.*** Even though the maximum quantile is not Pareto optimal in general, it does achieve good system throughput performance. If the rates achievable by users in a system are bounded, then maximum quantile scheduling is sum throughput optimal among policies that serve all users an equal fraction of time as the number of users increases. The following lemma is useful to prove this claim.

**Lemma 2.3.2.** *Consider a fixed saturated system with $n$ users, whose channel capacity variations satisfy Assumption 2.2.1 and served based on maximum quantile scheduling. Let $\epsilon, \delta \in (0,1)$, then there exists $n_{\epsilon,\delta}$ such that if $n > n_{\epsilon,\delta}$ at any slot where User $k$ gets scheduled for service, the user sees a rate exceeding $F_{X^k}^{-1}(1-\delta)$ with probability greater than $1-\epsilon$.*

*Proof.* As discussed in the previous section, whenever User $k$ gets served under maximum quantile scheduling, it sees a rate $F_{X^k}^{-1}(U^{(n)})$. In order to ensure the

desired condition is satisfied we require that

$$\Pr(F_{X^k}^{-1}(U^{(n)}) > F_{X^k}^{-1}(1-\delta)) > 1-\epsilon.$$

Since $F_{X^k}^{-1}(\cdot)$ is an increasing function, the above inequality can be rewritten as

$$\Pr(U^{(n)} > (1-\delta)) > 1-\epsilon.$$

From (2.4), we get

$$1-(1-\delta)^n > 1-\epsilon.$$

Simplifying and taking log, we get

$$n > \frac{\ln \epsilon}{\ln(1-\delta)}.$$

Defining

$$n_{\epsilon,\delta} = \lceil \frac{\ln \epsilon}{\ln(1-\delta)} \rceil,$$

we have that for any $n \geq n_{\epsilon,\delta}$, whenever User $k$ is served, it will experience a rate greater than $F_{X^k}^{-1}(1-\delta)$ with probability greater than $1-\epsilon$. □

The following theorem follows from Lemma 2.3.2 and formally states our claim.

**Theorem 2.3.3.** *Consider a fixed saturated system with n users, whose channel capacity variations satisfy Assumption 2.2.1 and are served using maximum quantile scheduling. Suppose each User i has a maximum instantaneous rate of $r_{max}^i < \infty$. Then as $n \to \infty$, each user is likely to be served at his maximum rate, so maximum quantile scheduling is sum throughput optimal.*

Summarizing, we observe that even though maximum quantile scheduling is not Pareto optimal, it is likely to give a good throughput performance.

## 2.4 Penalty due to Measurement

We now focus on the measurement aspects of opportunistic scheduling. We will first consider the throughput penalty incurred by maximum quantile scheduling due to estimation of rate distributions of users under fast fading, and present simulation results for the slow fading case. Following this, we will compare the penalty incurred by maximum quantile to that incurred by sum throughput optimal scheme in (2.2), via simulations.

### 2.4.1 Maximum Quantile Scheduling based on Empirical Distributions

Assumption 2.2.1 required that the channel capacity distribution, i.e., $F_{X^i}(\cdot)$ of each user be known at the base station. This is unlikely, and in this subsection we consider the penalty in throughput seen by users in a $n$ user fixed saturated system due to such mistakes by the scheduler.

Suppose the quantile of the current rate of a user is estimated using the previous $m$ samples of the user's rate. The empirical distribution of User $i$ during slot $t$ based on $m$ previous samples is denoted by $\tilde{F}^{m,t}_{X^i}(\cdot)$ and is given by

$$\tilde{F}^{m,t}_{X^i}(x) = \frac{1}{m}\sum_{j=1}^{m} 1\{X^i(t-j) \leq x\}. \tag{2.6}$$

Note that the above way of estimating is similar to the score function described in [4], however no attempt was made there to evaluate the penalty due to incorrect distribution estimation as function of $n$ and $m$.

Thus maximum quantile scheduling of users based on estimated distributions, would choose User $k(t)$ for service during slot $t$ if

$$k(t) \in \arg\max_{i=1,\ldots,n} \tilde{F}^{m,t}_{X^i}(x^i(t)),$$

with ties being broken arbitrarily.

Let us examine the properties of the above scheme. It can be shown that for any user on any slot $t$, $\tilde{F}^{m,t}_{X^i}(X^i(t))$ is uniformly distributed on $\{0, \frac{1}{m} \ldots, 1\}$.

Therefore, it is easy to see that even with estimated distributions, maximum quantile scheduling will still serve each user an equal fraction of time.

Calculating the penalty due to estimation seems to be intractable under slow fading, *therefore we add an additional assumption of fast fading, i.e., channel capacity realization of a user in a slot is independent across slots.* Even though fast fading users' channel capacity is not usually true, independence of samples can be roughly achieved by taking samples that are sufficiently apart in time or for some physical layer follows from system design, see e.g. 'opportunistic beamforming' [37]. The assumption is also likely to be true in OFDM based systems where slot times are relatively long.

We now calculate the long term throughput achieved by users under maximum quantile scheduling based on estimated distributions. Here, since we are interested in the stationary behavior, we simplify notation for the estimated distribution to $\tilde{F}^m_{X^i}(\cdot)$. Following theorem characterizes the performance of this scheme, a proof is given in Appendix 2.8.3.

**Theorem 2.4.1.** *Consider a fixed saturated system with n users whose channel capacity variations satisfy Assumption 2.2.1. Suppose the channel capacity distributions in such a system are estimated via (2.6) based on m* independent *samples of a user's channel and users are served using maximum quantile scheduling, then the long term throughput achieved by User k is given by*

$$\tilde{G}^k_{mq}(n,m) = \frac{E[F^{-1}_{X^k}(\tilde{U}_{n,m})]}{n},$$

*where $\tilde{U}_{n,m}$ is a continuous r.v. on $[0,1]$ having a probability density function*

$$f_{\tilde{U}_{n,m}}(u) = \sum_{j=0}^{m} \left( \begin{array}{c} m \\ j \end{array} \right) u^j (1-u)^{m-j} \frac{((j+1)^n - j^n)}{(m+1)^{n-1}}. \tag{2.7}$$

Recall that $R^i_{mq}$ represents the rate distribution seen by User $i$ when selected for service on a typical slot by maximum quantile scheduling (with perfect distribution knowledge). Let $\tilde{R}^{i,m}_{mq}$ denote the same quantity for maximum quantile scheduling when the distributions are estimated using $m$ samples.

We show that $\tilde{R}^{i,m}_{mq}$ and $R^i_{mq}$ are 'closely related' random variables, i.e., the rate seen by a user when served under empirical distributions is similar to that seen when the distributions are perfectly known. This is used to show that the average throughput achieved by a user when empirical distributions are used is less than or equal to that achieved when distributions are perfectly known, i.e., $\tilde{G}^k_{mq}(n,m) \leq G^k_{mq}(n)$ and bound the relative throughput penalty due to estimation. Our result is formally stated below, the proof given in Appendix 2.8.4.

**Theorem 2.4.2.** *Consider a fixed saturated system with n users whose channel capacity variations satisfy Assumption 2.2.1. Then under fast fading* $\forall n, m$,

$$(\frac{m+1}{n}(1-(\frac{m}{m+1})^n)) \leq \frac{\Pr(\tilde{R}^{i,m}_{mq} \leq r)}{\Pr(R^i_{mq} \leq r)} \leq 1, \ \forall r,$$

*and*

$$G^k_{mq}(n) \geq \tilde{G}^k_{mq}(n,m), \ \forall m,$$

*and the relative throughput penalty is bounded by*

$$\frac{|G^k_{mq}(n) - \tilde{G}^k_{mq}(n,m)|}{G^k_{mq}(n)} \leq 1 - \frac{m+1}{n}(1-(\frac{m}{m+1})^n).$$

Note that the above theorem can be extended to the discrete rate case, i.e., where $X^i$'s are discrete by modifying the system model. In the modified system model, each user feeds back the estimated quantile of its current channel quality along with its current supported rate to the base station in every slot. The estimated quantile is based on $m$ previous independent samples of a *continuous* measure of the channel quality, e.g., signal to noise ratio (SNR). The base station chooses to serve the user with the highest estimated quantile (that it has received from the feedback).

To understand the scaling of the number of independent samples $m$ required to limit the throughput penalty, note that for a reasonably large $n$,

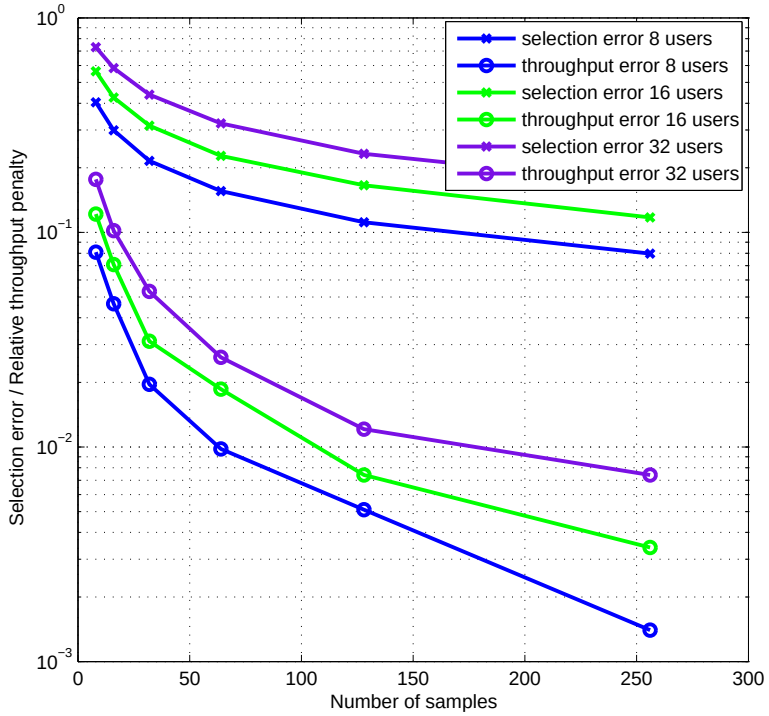



Figure 2.4: The top three curves plot the selection error probability for maximum quantile scheduling, due to estimated distributions with increasing number of users. The bottom three curves plot the relative throughput penalty for the same.

if $m$ scales linearly with $n$, then

$$(\frac{m}{m+1})^n = (1+\frac{1}{m})^{-n} \approx e^{-\frac{n}{m}}.$$

Expanding $e^{-\frac{n}{m}}$ and simplifying, we get that the penalty is equal to

$$1 - \frac{m+1}{m} + \frac{m+1}{n}(\frac{1}{2}(\frac{n}{m})^2 - \ldots),$$

which is upper bounded by $\frac{n}{2m}$. Therefore to achieve a relative error less than $\epsilon$, approximately $\frac{n}{2\epsilon}$ samples are needed. For example to achieve an error less than 5%, approximately $10n$ samples are needed. Therefore for a given error bound, the number of samples required will at worst grow roughly linearly with the number of users contending.

To validate these results, we ran some simulations. The simulation setup is the same as discussed in Section 2.2 except that the channel capacity variation for all users fast fading across slots (in accordance with the assumption used while developing the bound). We observed the throughput penalty for different values of $n$ and $m$. The value of $n$ is varied from 8 to 16 to 32, while $m$ is varied by a factor of 2 from 8 to 256 for a given value of $n$. As shown in Figure 2.4, the bound is clearly met, in fact the results indicate that our bound is quite conservative (which is not surprising, since the bound is distribution free). For example, a penalty of around 1% is achieved with only 64 samples for 8 users, whereas the bound suggests 5%.

We also plot the selection error probability in the figure, i.e., the fraction of slots where the user selected with maximum quantile is *not* chosen due to error in estimation of distribution. As the plot indicates, this can be quite high. Simple analysis can be used to show that the number of samples required to achieve a given error probability grows roughly as $O(n^2)$. Therefore, even though mistakes may be made in selecting the user with the highest quantile, the throughput penalty in making an error is not large.

Let us consider the relevance of the bound under slow fading. The need for $m$ independent samples immediately suggests the need for sampling $m$ coherence time intervals to achieve the required penalty. We ran simulations to confirm this conjecture. The simulation consisted of two (earlier described) classes of slow Rayleigh fading users with 5 users each, we aimed for a throughput penalty of 5%. The Doppler spread for the channels was varied from 10 Hz to 50 Hz in steps of 10 Hz. Let $f_D$ denote the Doppler spread, then the coherence time can be estimated using the formula $\frac{9}{16\pi f_D}$ [28]. Given the coherence time, the total number of users and the required penalty, the number of slots needed to estimate the rate distributions can be ascertained. The simulation results are plotted in Figure 2.5, as can be observed, the required penalty is easily met in all cases.

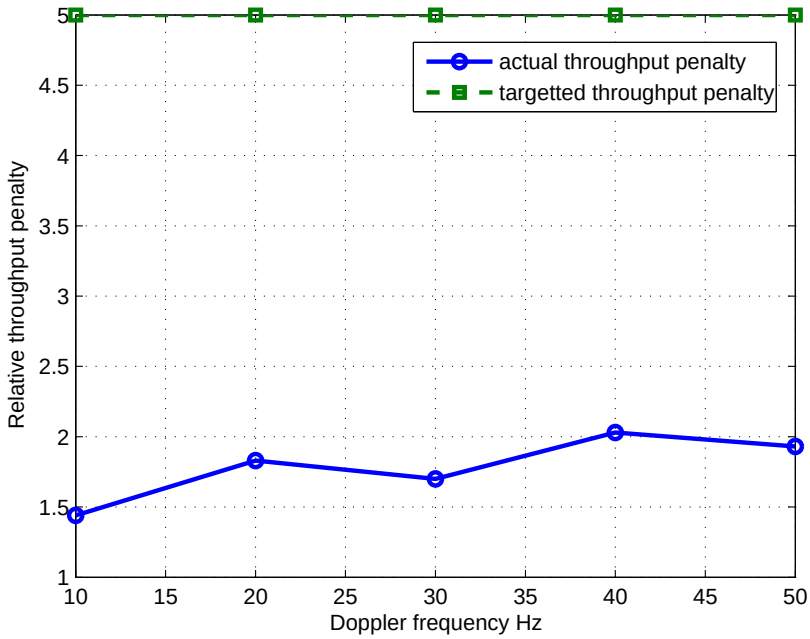



Figure 2.5: Relative throughput penalty for 10 users with slow Rayleigh fading channel capacities.

Note that in our simulations we found that for Doppler spread of 10 Hz, 932 slots were needed. (Other Doppler spreads required 466, 311, 233, 187 slots.) This corresponds to 1.55 seconds (slot size is 1.67 msec), it may be reasonable to expect the system to be stationary for such a period because the Doppler spread is quite low, i.e., users/objects are moving quite slowly. In other words, even though very slowly fading systems may require a large number of samples to achieve the desired penalty, it may also be reasonable to expect such channels to be stationary over large periods of time.

***Discussion of the bound.*** Theorem 2.4.2 has several interesting implications, which we discuss below.

- The bound shows that the throughput penalty due to estimation of users' distributions can be bounded for *any* distribution.

- The theorem is strong in the sense that it shows a relationship between

distributions of rates seen by the user in both the empirical and perfectly known distribution cases.

- Furthermore, the number of samples needed to achieve small penalty is *only linear in the number of users*. This is fairly limited (at least for the fast fading case) because the slot sizes are usually of the order of milliseconds.

- The dependence of penalty only on the number of users is significant, because this allows the bound to extend to the unsaturated and dynamic regime. To achieve a certain penalty, a system designer only needs to estimate the 'average' number of users that will be competing for service at any given time, and not on the users' distribution or traffic characteristics. We reiterate here that it is difficult to even design heuristics to redefine weights in dynamic and unsaturated scenarios for other weight based schemes.

- The dependence on only the number of users also allows the theorem to extend to quasi stationary rate distributions. Since one requires only $O(n)$ slots (under fast fading) to limit the penalty due to measurement, therefore $O(n)$ slots after a user's distribution changes, the system will not experience any penalty due to change in distribution. Thus we conjecture that if users' channels are stationary for roughly $O(n^2)$ slots (with appropriate constants) then the penalty due to changes in user's rate distributions will be negligible.

Summarizing, maximum quantile scheduling under estimated distribution case is not only fair and suffers from fairly limited penalty, but is quite easy to design and implement in a practical scenario.

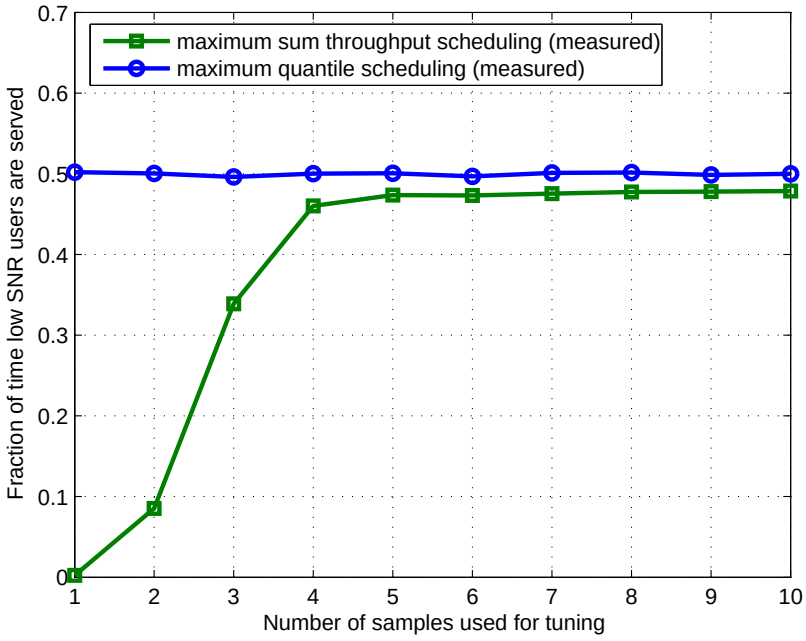



Figure 2.6: Fraction of time low SNR users are served by measurement based maximum sum throughput optimal and maximum quantile scheduling schemes, with increasing number of tuning samples.

### 2.4.2 Throughput Penalty Comparisons

Recall that if the users' weights $\nu^i$ are properly set in (2.2), then the scheme maximizes sum throughput under temporal fairness. However in practice the weights for each user need to be estimated. Let us investigate the sensitivity of system throughput to errors in these weights by performing two controlled experiments. We will use the previously discussed experimental setup, however again in accordance with the assumption on the bound developed, in this subsection we will assume that the channel capacity of all users is fast fading.

In the first experiment, there are 5 users in each class, and the weights $\nu^i$ for all users are initialized to 0. We train the weights for $m$ slots according to the stochastic approximation algorithm suggested in [19], and observe the average penalty in performance due to errors in weights on the $(m+1)^{st}$ slot.

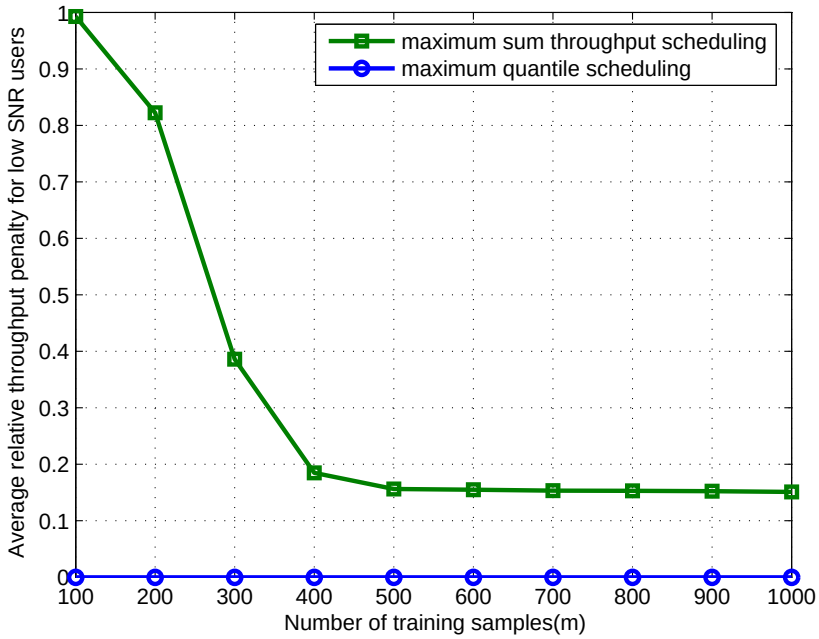



Figure 2.7: Average relative throughput penalty incurred by the class of low SNR users for increasing number of tuning samples.

We refer the reader to [19] for details on the training algorithm. We evaluate two performance parameters, the fraction of time low SNR users are served, and the relative penalty in throughput achieved by those users as compared to that achieved when weights are perfectly known.

The stochastic approximation algorithm for estimating the $\nu^i$'s has several parameters $(w, \delta, \delta_i)$ that need to be set, we first set these parameters equal to those suggested in [19]. However, the scheduling scheme served the low SNR user less than 0.1% of time even when $m = 2000$ (again demonstrating that measurement based weights may severely affect performance). We changed the parameters to $w = 0.005$, $\delta = 0.2$ and $\delta_i = 0.1$, which exhibited better performance.

Figure 2.6 shows the fraction of time low SNR users are served as an increasing number of training samples $m$ is used. We also plot the corresponding results for maximum quantile scheduling. Note that maximum quantile

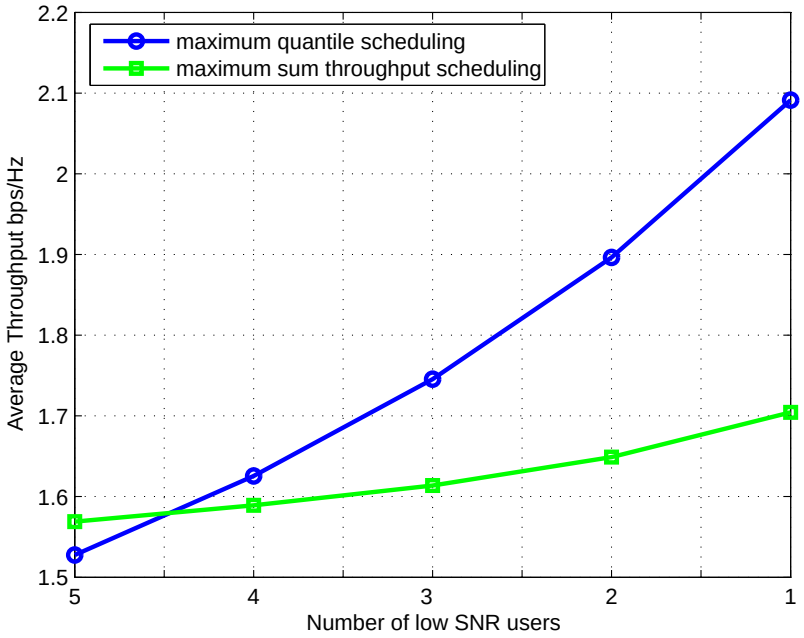



Figure 2.8: Throughput achieved by maximum sum throughput under temporal fairness and maximum quantile scheduling with decreasing number of low SNR users.

scheduling always serves low SNR users close to 0.5 fraction of time. By contrast, maximum sum throughput takes around 400 samples to converge to approximately 0.47 and then shows negligible improvement. This is because the granularity of training is not sufficiently small, however as suggested in the previous paragraph, if one reduces these updates, then the convergence time may be much larger.

Figure 2.7 shows the throughput penalty for the low SNR users for an increasing number of training samples $m$. While the throughput penalty is virtually 0 under maximum quantile scheduling, note that there is penalty of 15% even for 1000 training samples. Therefore a 6% loss in temporal fairness can lead to a 15% loss in throughput.

In the second experiment, suppose there are initially 5 users belonging to each class, with estimates for $\nu^i$ converged to their true values. Now if a user leaves the system the values of weights would have to change, so if the

system does not tune fast enough, then the maximum sum throughput scheme may incur a throughput penalty. We simulated the throughput achieved by the scheme under the previously converged values of weights and compare it to that achieved by maximum quantile scheduling with distribution estimates converged.

Figure 2.8 shows the throughput achieved by both the schemes when the number of low SNR users is reduced. Note that the throughput difference between the two schemes is small even when both classes have 5 users each. Then if the number of low SNR users goes from 5 to 4, maximum quantile scheduling immediately starts doing better. We observed a similar trend when the high SNR users were reduced.

## 2.5 Performance in Fixed Unsaturated Regime

In this section, we consider a fixed unsaturated system. One can show that the throughput achieved by an infinitely backlogged User $k$ in an unsaturated system is lower bounded by $G^k_{mq}(n)$, i.e., the throughput achieved in a fixed saturated system. However, the way in which resources are allocated impacts the delay for e.g. real-time traffic. Therefore, we will evaluate the packet delay in this section.

In our simulations, we compare the performance of maximum quantile scheduling with maximum rate, proportionally fair and the exponential rule. We do not compare the performance with the maximum sum throughput scheme (2.2), because it is unclear how to set the weights for this scheme in an unsaturated scenario.

Our setup is the same as before with each user experiencing Rayleigh slow fading with either mean SNR of 2 or 0.1 at a Doppler frequency of 15Hz. There are 15 users in each class. All users have Poisson packet arrivals with equal average arrival rate. Each packet is 1500 bytes. Packet delay for a packet is measured by finding the difference between packet arrival time and

the time when the packet has been *completely* transmitted. We set all the weights equal to 1 in the exponential rule (We also experimented by weighting a user's queue inversely proportional to its channel mean, but that increased the average delay for the exponential rule.) We assume that the distribution is perfectly known at the scheduler for maximum quantile scheduling, i.e., the estimates of the distributions have converged (this may be reasonable for fixed systems).

Figure 2.9 shows the average packet delay across users as the load increases. (Maximum rate scheduling has the worst performance among all schemes, for simplicity its plot has not been shown). Maximum quantile always has the lowest packet delay, and achieves around 50% reduction in packet delay (at total arrival rate of 29 packets per second) as compared to proportionally fair. This is surprising, since unlike exponential rule, maximum quantile is completely *insensitive* to queue lengths. This underscores the importance of scheduling according to opportunism, rather than simply the rate and/or queue lengths.

## 2.6 Performance in Dynamic Saturated Regime

In this section, we study the dynamic saturated system. We will first analyze the performance of maximum quantile scheduling scheme for certain channel models and then compare its performance to maximum rate, proportional fair and maximum sum throughput under temporal constraints. Note that the exponential rule does not make sense in a saturated scenario. Dynamic saturated system is a good model for a base station supporting file transfers, therefore, a good metric for performance here is the average file transfer delay.

### 2.6.1 Multiclass Processor Sharing Model

In [7], the delay performance of a version of proportionally fair scheduling is studied by modelling dynamic systems as a multiclass $M/GI/1$ processor

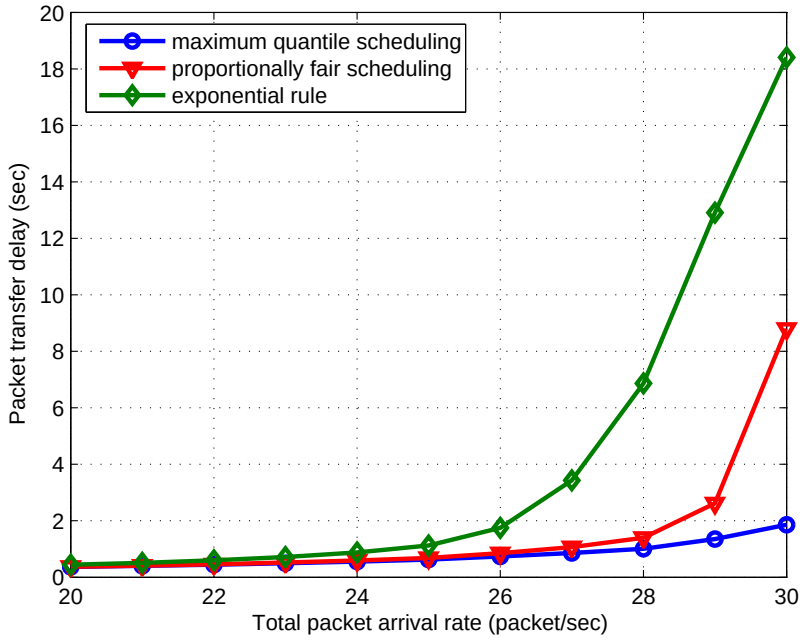



Figure 2.9: Packet delay performance of maximum quantile, proportionally fair and exponential rule scheduling.

sharing queue for a certain class of channel models. In this subsection we do the same for maximum quantile scheduling, to provide a tool to design and evaluate the performance of such base stations. In this subsection, we will deal with classes of users rather than individual users, i.e., $X^i$ will refer to rate distribution associated with Class $i$ instead of User $i$. All users belong to the same class will have i.i.d. rate distributions.

Let us first describe the multiclass $M/GI/1$ processor sharing model [3]. In the model, all users are served an equal fraction of time, with users belonging to the same class getting served at equal rate. For example all users in Class $i$ are served at an average rate of $G^i_{mq}(n)$ under maximum quantile scheduling, when there are a total of $n$ users in the system. The use of the mean rate is motivated by the time scale separation results see e.g. [24][25]. User arrival to each class is according to a Poisson process, with possibly different loads for each class. Each user has a single file associated with it.

The size of users' files in a class can be distributed according to any finite mean distribution.

In general, it is not possible to find the exact performance for maximum quantile scheduling because the throughput (or service) formula for maximum quantile scheduling does not satisfy the 'balance property'[5], i.e.,

$$\frac{G^i_{mq}(n+1)}{G^i_{mq}(n)} \neq \frac{G^j_{mq}(n+1)}{G^j_{mq}(n)},$$

for any two Classes $i$ and $j$. This does not allow one to derive the explicit delay performance of a user. However, one can show that service has a monotonicity property, i.e., for any Class $i$, $G^i_{mq}(n+1) < G^i_{mq}(n)$ (we omit the proof here). In other words, whenever a user leaves the system, the service rate of all other users necessarily increases. This allows one to find lower and upper bounds for the performance [5].

Using the monotonicity, one can show that for any Class $i$ ,

$$1 \leq \frac{E[X^{i,(n+1)}]}{E[X^{i,(n)}]} \leq 1 + \frac{1}{n}.$$

This can be used to develop lower and upper bounds on delay performance. However, note that these bounds hold for any rate distribution, and so are quite loose, therefore we attempt to find better bounds.

Consider the case, where the inverse distribution rate function of users belonging to Class $i$ has the following form

$$F^{-1}_{X^i}(u) = c^i u^{\alpha^i},\ u \in [0,1],$$

here both $c^i$ and $\alpha^i$ are positive and finite constants. Then,

$$E[X^{i,(n)}] = c^i \int_0^1 F^{-1}_{X^i}(u) n u^{n-1} du = \frac{c^i n}{n+\alpha^i}.$$

Therefore, it is easy to see that

$$\frac{E[X^{i,(n+1)}]}{E[X^{i,(n)}]} = 1 + \frac{\alpha^i}{n(n+\alpha^i+1)}.$$

Let $n_c$ be the total number of Classes in the system. Then, note that

$$1 + \frac{\alpha_{min}}{n(n + \alpha_{min} + 1)} \leq \frac{E[X^{i,(n+1)}]}{E[X^{i,(n)}]} \leq 1 + \frac{\alpha_{max}}{n(n + \alpha_{max} + 1)},$$

where $\alpha_{min} := \min_{i \in n_c} \alpha^i$ and $\alpha_{max} := \max_{i \in n_c} \alpha^i$. Therefore the real system's service can be lower bounded by a system where all classes of users have inverse distribution rate function of the form $F^{-1}_{X^i}(u) = c^i u^{\alpha_{min}}$. We call this the lower service bound. Similarly an upper service bound can be developed with $\alpha_{max}$.

Note that $\frac{d}{d\alpha^i}(1 + \frac{\alpha^i}{n(n+\alpha^i+1)}) = \frac{n+1}{n(n+\alpha^i+1)^2}$, i.e, the ratio changes quite slowly with $\alpha^i$, this indicates that the bound developed may be reasonably tight.

We now define some additional notation. Let $\lambda^i$ be the average arrival rate of users belonging to Class $i$ and let $Z^i$ be the random variable representing the length of file associated with users. Then the *normalized* load offered by Class $i$ can be defined as $\rho^i := \frac{\lambda^i E[Z^i]}{E[X^i]}$. (Note that by normalizing by $E[X^i]$, we capture the heterogeneity across classes.) Therefore, the total load offered to the system is equal to $\rho := \sum_{i=1}^{n_c} \rho^i$. Define

$$\phi(\alpha, n) := \prod_{k=1}^{n} \prod_{j=1}^{k} (1 + \frac{\alpha}{j(j + \alpha + 1)}).$$

Then, from [7], the probability of $n$ users in the system for the lower service bound $\pi(n, \alpha_{min})$ is given by

$$\pi(n, \alpha_{min}) = [\sum_{n=0}^{\infty} \frac{\rho^n}{\phi(\alpha_{min}, n)}]^{-1} \frac{\rho^n}{\phi(\alpha_{min}, n)}.$$

By replacing $\alpha_{min}$ by $\alpha_{max}$, one can similarly find the probability for the upper service bound. The probability can be used to find the average number of users in the system and then Little's law can be used to find the average delay. One can show that the average delay for lower service bound is the *upper* average delay bound for the real system and vice versa for the upper service bound.

The above developed bounds are also valid for inverse rate distributions the form $F^{-1}_{X^i}(u) = \sum_{j=1}^{j^i} c^i_j u^{\alpha^i_j}$, $u \in [0, 1]$. Here $c^i_j$'s and $\alpha^i_j$'s are positive

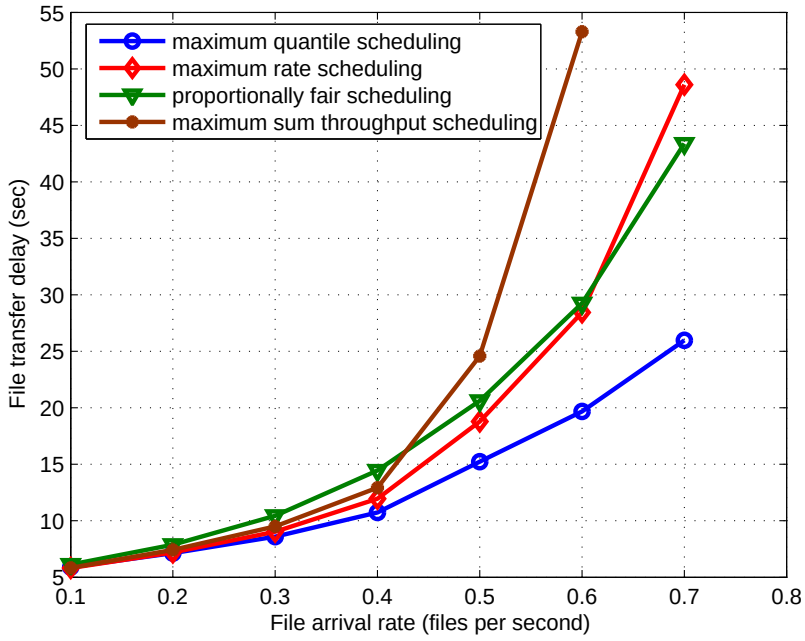



Figure 2.10: File transfer delay performance of maximum quantile, maximum rate, proportionally fair and maximum sum throughput scheduling.

constants. It seems that a large class of distributions can be approximated using this form to give a reasonable tool for performance evaluation.

We numerically calculated the bounds for sample systems. Our results indicate that the bounds are tight for low to medium loads, but poor for high loads.

### 2.6.2 Delay Performance Comparison

We now present some simulation results, again our setup is the same as before, however since the system is dynamic, the number of users will change with time. Users arrive to the system according to a Poison process, and are equally likely to belong to one of the two classes (where each class is experiencing slow Rayleigh fading). Each user has a file associated with it. The file sizes are exponentially distributed with a mean size of 60KB. We keep track of the time taken from a user's arrival to departure. For maximum

quantile, estimate for users rate distributions are generated by keeping track of previous samples. While the weights for maximum sum throughput under temporal fairness are trained using the stochastic approximation algorithm referred to in Section 2.4, with the values of the parameters same as before.

The average file transfer delay experienced by users is plotted with increasing load in Figure 2.10. The number of samples used for estimating users' distributions is 100 at a load of 0.1. This was increased linearly by 100 samples for every load increase of 0.1. As can be seen in the figure, maximum quantile scheduling outperforms both maximum rate and maximum sum throughput. In fact the reduction in delay is more than 40% at a load of 0.7. This again underscores the importance of scheduling according to the quantile. Also note that due to non convergence of weights, maximum sum throughput has the worst delay performance.

## 2.7 Conclusion

In summary we have evaluated measurement based opportunistic scheduling schemes from various perspectives and under various system regimes, e.g., dynamic/fixed, saturated/unsaturated. The key take away, is that, perhaps surprisingly, maximum quantile scheduling which would require estimation of each users channel rate distribution, realizes excellent performance, relative to proportionally fair, the exponential rule, and schemes that are optimal in terms of sum throughput subject to fairness. The main reason is that maximum quantile places systematic emphasis on scheduling users when they are high relative to their own distribution, while achieving temporal fairness. By contrast other schemes measure the degree to which fairness is achieved and bias scheduling decisions to compensate for biases. This compromises opportunism and also performance. Although the estimation of users distributions seems fairly straightforward and would be necessary to enable resource management and call admission decisions at a wireless point, the question remains

as to whether the additional complexity over simple schemes such as proportionally fair is warranted.

## 2.8 Appendix

### 2.8.1 Proof of Lemma 2.2.2

*Proof.* Let us consider the $i^{th}$ user. Clearly $U^i \geq 0$, and the maximum value of $F_{X^i}(X^i)$ is 1, therefore the range space of $U^i$ is restricted to $[0,1]$. For $\alpha \in [0,1]$ our goal is to show that

$$\Pr(U^i \leq \alpha) = \Pr(F_{X^i}(X^i) - p^i(X^i)W^i \leq \alpha) = \alpha.$$

Define $x_{j(\alpha)}$ as $x_{j(\alpha)} = F^{-1}_{X^i}(\alpha)$. Note that the index $j(\alpha) \in \{1, \ldots, l\}$ and $x_{j(\alpha)}$ is one of the $l$ discrete rates the users can support. Furthermore, the rates supported are increasing in their index so it follows that for all $j < j(\alpha)$,

$$F_{X^i}(x_j) < \alpha.$$

Since $p^i(X^i)W^i \geq 0$, it follows that if $j < j(\alpha)$ then $U^i \leq \alpha$. If $X^i = x_{j(\alpha)}$, then in order for $U^i \leq \alpha$ it must be the case that

$$p^i(x_{j(\alpha)})W^i \geq F_{X^i}(x_{j(\alpha)}) - \alpha,$$

Finally if $j > j(\alpha)$, then clearly $U^i > \alpha$. In summary we have that $\Pr(U^i \leq \alpha)$ is equal to

$$\Pr(X^i < x_{j(\alpha)}) + p^i(x_{j(\alpha)}) \Pr(p^i(x_{j(\alpha)})W^i \geq F_{X^i}(x_{j(\alpha)}) - \alpha).$$

This can be rewritten as

$$\sum_{j:j<j(\alpha)} p^i(x_j) + p^i(x_{j(\alpha)}) Pr(W^i \geq \frac{F_{X^i}(x_{j(\alpha)}) - \alpha}{p^i(x_{j(\alpha)})}). \tag{2.8}$$

Since $W^i$ is uniform on $[0,1]$,

$$Pr(W^i \geq \frac{F_{X^i}(x_{j(\alpha)}) - \alpha}{p^i(x_{j(\alpha)})}) = 1 - \frac{F_{X^i}(x_{j(\alpha)}) - \alpha}{p^i(x_{j(\alpha)})}.$$

Simplifying, we get

$$Pr(W^i \geq \frac{F_{X^i}(x_{j(\alpha)}) - \alpha}{p^i(x_{j(\alpha)})}) = \frac{\alpha - \sum_{j:j<j(\alpha)} p^i(x_j)}{p^i(x_{j(\alpha)})}.$$

Replacing $\Pr(W^i \geq \frac{F_{X^i}(x_{j(\alpha)})-\alpha}{p^i(x_{j(\alpha)})})$ in (2.8) and simplifying, we get

$$\Pr(U^i \leq \alpha) = \alpha.$$

Therefore $U^i, i = 1, \ldots, n$ are uniformly distributed on $[0, 1]$ and by Assumption 2.2.1 they are independent. □

### 2.8.2 Proof of Theorem 2.3.1

*Proof.* Define $U_\beta := \sum_{i=1}^n U^i \mathbf{1}_{S^i_\beta}$, i.e., the total opportunism achieved by $\beta$. Let $U^i_\beta = U^i | S^i_\beta$, i.e., the quantile of User $i$ conditioned on getting served by $\beta$. Then

$$\Pr(U_\beta > u) = \sum_{i=1}^n \Pr(U^i_\beta > u) \Pr(S^i_\beta),\ u \in [0, 1].$$

Let $j(u) = \arg\min_{i=1,\ldots,n} \Pr(U^i_\beta > u)$. Since $\beta$ is non idling, $\sum_{i=1}^n \Pr(S^i_\beta) = 1$, so

$$\Pr(U_\beta > u) \geq \Pr(U^{j(u)}_\beta > u).$$

Recall that $U^{(n)}$ is the maximum of $n$ i.i.d. uniformly distributed random variables, then since $\beta$ is distinct, there must be a $u'$ such that

$$\Pr(U^{(n)} > u') > \Pr(U_\beta > u') \geq \Pr(U^{j(u')}_\beta > u').$$

Let $U^i_{mq}$ be the same quantity as $U^i_\beta$ for maximum quantile scheduling. Now recall that under maximum quantile scheduling, a User $i$ is selected for service only when its quantile is the highest, i.e., $U^i_{mq} \sim U^{(n)}$. Then

$$\Pr(U^{j(u')}_{mq} > u') > \Pr(U^{j(u')}_\beta > u').$$

So $U^{j(u')}_\beta \not\geq^{st} U^{j(u')}_{mq}$. Note that for any User $i$, $R^i_\beta = F^{-1}_{X^i}(U^i_\beta)$ and $R^i_{mq} = F^{-1}_{X^i}(U^i_{mq})$. Now since $F^{-1}_{X^{j(u')}}(\cdot)$ is an increasing function, then

$$R^{j(u')}_\beta \not\geq^{st} R^{j(u')}_{mq}.$$

□

### 2.8.3 Proof of Theorem 2.4.1

*Proof.* Recall that $S^k$ is the event denoting the selection of User $k$ for service. Since each user is equally likely to be served, $\Pr(S^k) = \frac{1}{n}$, and

$$\tilde{G}^k_{mq}(n,m) = E[X^k|S^k]\Pr(S^k) = \frac{E[X^k|S^k]}{n}.$$

Let us now evaluate $E[X^k|S^k]$ by conditioning on $\tilde{F}^m_{X^k}(X^k)$, we have that

$$E[X^k|S^k] = \sum_{j=0}^{m} E[X^k|S^k, \tilde{F}^m_{X^k}(X^k) = \frac{j}{m}]\Pr(\tilde{F}^m_{X^k}(X^k) = \frac{j}{m}|S^k).$$

Note that the selection of a user in a slot is independent of its current rate, given its estimated current quantile, so

$$E[X^k|S^k, \tilde{F}^m_{X^k}(X^k) = \frac{j}{m}] = E[X^k|\tilde{F}^m_{X^k}(X^k) = \frac{j}{m}].$$

Since $\tilde{F}^m_{X^k}(X^k)$ are uniformly distributed on $\{0, \frac{1}{m}, \ldots, 1\}$ and ties are broken randomly,

$$\Pr(\tilde{F}^m_{X^k}(X^k) = \frac{j}{m}|S^k) = \frac{(j+1)^n - j^n}{(m+1)^n}.$$

Now consider $E[X^k|\tilde{F}^m_{X^k}(X^k) = \frac{j}{m}]$, by using Bayes' formula and the fact that $\left( \begin{array}{c} m \\ j \end{array} \right) \int_0^1 y^j(1-y)^{m-j}dy = \frac{1}{m+1}$, one can show that

$$E[X^k|\tilde{F}^m_{X^k}(X^k) = \frac{j}{m}] =$$
$$(m+1)\left( \begin{array}{c} m \\ j \end{array} \right) \int_0^\infty x(F_{X^k}(x))^j(1-F_{X^k}(x))^{m-j} f_{X^k}(x)dx,$$

where $f_{X^k}(\cdot)$ is the probability density function of the SNR associated with User $k$. Now using a change of variables this can be rewritten as

$$E[X^k|\tilde{F}^m_{X^k}(X^k) = \frac{j}{m}] = (m+1)\left( \begin{array}{c} m \\ j \end{array} \right) \int_0^1 F^{-1}_{X^k}(u)u^j(1-u)^{m-j}du.$$

So it follows that $\tilde{G}^k_{mq}(n,m)$ is given by

$$\frac{1}{n}\int_0^1 F^{-1}_{X^k}(u)(\sum_{j=0}^{m}\left( \begin{array}{c} m \\ j \end{array} \right) u^j(1-u)^{m-j}\frac{((j+1)^n - j^n)}{(m+1)^{n-1}})du.$$

This completes the proof. □

### 2.8.4 Proof of Theorem 2.4.2

We present a few useful lemmas before proving Theorem 2.4.2.

**Lemma 2.8.1.** *Let $H$ be a binomial r.v. with parameters $(m, u)$. Consider the moment generating function of $H$, $M(s) := (1 - u + ue^s)^m$. Its $l^{th}$ derivative is given by*

$$\frac{d^l M(s)}{ds^l} = \sum_{j=1}^{l} b_{j,l} \frac{m!}{(m-j)!}(1 - u + ue^s)^{m-j}(ue^s)^j. \tag{2.9}$$

*Here $b_{j,l}$'s are constants with the following properties:*

- $b_{1,1} = 1$
- $b_{j,l} = jb_{j,l-1} + b_{j-1,l-1}$, $\forall j = 1, \ldots, l$, $\forall l$
- $b_{0,l} = b_{l+1,l} = 0$, $\forall l$.

*Note that since $b_{1,1} = 1$ and $b_{l+1,l} = 0$, $\forall l$, from the second property one can show that $b_{l,l} = b_{l-1,l-1} = 1$, $\forall l$.*

*Proof.* The lemma clearly holds for $l = 1$. We give a proof by induction on $l$. Assume the lemma holds for $l$, i.e., (2.9) is true. Then, to prove the lemma for $l + 1$, we differentiate (2.9) and after some rearrangement get

$$\frac{d^{l+1} M(s)}{ds^{l+1}} = \sum_{j=1}^{l+1} [(jb_{j,l} + b_{j-1,l}) \frac{m!}{(m-j)!}(1 - u + ue^s)^{m-j}(ue^s)^j].$$

This completes the proof. □

From Lemma 2.8.1 it follows that the $l^{th}$ order moment of $H$ is given by

$$E[H^l] = \sum_{j=1}^{l} b_{j,l} \frac{m!}{(m-j)!} u^j. \tag{2.10}$$

The following lemma exhibits an inequality between the moments of $H$.

**Lemma 2.8.2.** *Let $H$ be a binomial r.v. with parameters $(m, u)$. Then for all $l$ such that $l \leq m$,*

$$E[H^{l+1}] \leq (mu + l(1-u))E[H^l]. \tag{2.11}$$

*Proof.* The right side of (2.11) can be expressed as

$$((m-l)u + l)E[H^l].$$

Using (2.10), the above equation can be rewritten as

$$\frac{m!}{(m-l-1)!}u^{l+1} + \sum_{j=1}^{l}[lb_{j,l}\frac{m!}{(m-j)!} + (m-l)b_{j-1,l}\frac{m!}{(m-j+1)!}]u^j. \tag{2.12}$$

If one splits $lb_{j,l}\frac{m!}{(m-j)!}$ in the following way

$$lb_{j,l}\frac{m!}{(m-j)!} = jb_{j,l}\frac{m!}{(m-j)!} + (l-j)b_{j,l}\frac{m!}{(m-j)!},$$

then (2.12) in turn can be expressed as

$$\frac{m!}{(m-l-1)!}u^{l+1} + \sum_{j=1}^{l}[(jb_{j,l}\frac{m!}{(m-j)!} + (m-l)b_{j-1,l}$$
$$\frac{m!}{(m-j+1)!})u^j + (l-j+1)b_{j-1,l}\frac{m!}{(m-j+1)!}u^{j-1}].$$

Now since $0 \leq u \leq 1$, then $\forall j$, $u^{j-1} \leq u^j$. So, from the above equation we get

$$(mu + l(1-u))E[H^l] \geq \frac{m!}{(m-l-1)!}u^{l+1} + \sum_{j=1}^{l}[(jb_{j,l}\frac{m!}{(m-j)!}$$
$$+(m-l)b_{j-1,l}\frac{m!}{(m-j+1)!}) + (l-j+1)b_{j-1,l}\frac{m!}{(m-j+1)!}]u^j.$$

Combining the last two terms in the summation of the above inequality, we get

$$(mu + l(1-u))E[H^l] \geq \frac{m!}{(m-l-1)!}u^{l+1} + \sum_{j=1}^{l}(jb_{j,l} + b_{j-1,l})\frac{m!}{(m-j)!}u^j$$

This proves (2.11). □

Next we show that $U^{(n)}$ dominates $\tilde{U}_{n,m}$ in a likelihood ratio ordering sense, i.e., $U^{(n)} \geq^{lr} \tilde{U}_{n,m}$ [20][29][21]. This is a strong form of dominance which means that $f_{U^{(n)}}(u)/f_{\tilde{U}_{n,m}}(u)$ is non decreasing in $u$, or $f_{\tilde{U}_{n,m}}(u)/f_{U^{(n)}}(u)$ is non increasing in $u$ (here $f_{U^{(n)}}(u)$ is the probability density function of $U^{(n)}$). If $U^{(n)} \geq^{lr} \tilde{U}_{n,m}$, it follows that $U^{(n)} \geq^{st} \tilde{U}_{n,m}$.

**Lemma 2.8.3.** *For the random variables $U^{(n)}$ and $\tilde{U}_{n,m}$ given by (2.4) and (2.7) respectively, then $\forall n, m$ $U^{(n)} \geq^{lr} \tilde{U}_{n,m}$.*

*Proof.* To prove the lemma, we need to show

$$\frac{d}{du}\left[\frac{f_{\tilde{U}_{n,m}}(u)}{f_{U^{(n)}}(u)}\right] \leq 0,$$

$\forall u \in (0,1]$. To prove this, it is sufficient to show

$$f_{U^{(n)}}(u)\left[\frac{df_{\tilde{U}_{n,m}}(u)}{du}\right] - f_{\tilde{U}_{n,m}}(u)\left[\frac{df_{U^{(n)}}(u)}{du}\right] \leq 0.$$

Note that $f_{U^{(n)}}(u) = nu^{n-1}$. Then expanding, we get

$$\frac{1}{(m+1)^{n-1}}[nu^{n-1}(-m(1-u)^{m-1} + \sum_{j=1}^{m-1}\left(\begin{array}{c} m \\ j \end{array}\right) u^{j-1}(1-u)^{m-j-1}(j-mu)((j+1)^n - j^n) + mu^{m-1}((m+1)^n - m^n)) - n(n-1)u^{n-2}(\sum_{j=0}^{m}\left(\begin{array}{c} m \\ j \end{array}\right) u^j(1-u)^{m-j}((j+1)^n - j^n))] \leq 0.$$

Simplifying and multiplying both sides by $(1-u)$, we get

$$(-mu(1-u)^m + \sum_{j=1}^{m-1}\left(\begin{array}{c} m \\ j \end{array}\right)(j-mu)u^j(1-u)^{m-j}((j+1)^n - j^n) + (m-mu)u^m((m+1)^n - m^n)) - (n-1)(1-u)(\sum_{j=0}^{m}\left(\begin{array}{c} m \\ j \end{array}\right) u^j(1-u)^{m-j}((j+1)^n - j^n)) \leq 0.$$

The above inequality can be rewritten as

$$\sum_{j=0}^{m} \left( \begin{array}{c} m \\ j \end{array} \right) (j - mu - (n-1)(1-u))u^j(1-u)^{m-j}((j+1)^n - j^n) \leq 0.$$

Then the inequality clearly holds for $m < n$. However the more interesting case is when $m \geq n$, and this requires a few more steps. Note that $\left( \begin{array}{c} m \\ j \end{array} \right) u^j(1-u)^{m-j}$ is the probability that a binomial r.v. with parameter $(m, u)$ has a value $j$, i.e., the same as that of $H$. Then the inequality can be rewritten in terms of expectations as

$$E[(H - mu)((H+1)^n - H^n)] - (n-1)(1-u)E[(H+1)^n - H^n] \leq 0.$$

This can be further rewritten as

$$E[H((H+1)^n - H^n)] \leq (mu + (n-1)(1-u))E[(H+1)^n - H^n]. \quad (2.13)$$

Expanding $(H+1)^n$ and simplifying, one can show that (2.13) will hold if

$$E[H^{l+1}] \leq (mu + l(1-u))E[H^l],$$

$\forall l < n \leq m$. This follows from Lemma 2.8.2. This completes the proof. □

We now prove Theorem 2.4.2.

*Proof.* To prove the first claim, define $u := F_{X^i}(r)$ and consider

$$F_{U^{(n)}}(u) - F_{\tilde{U}_{n,m}}(u), \ \forall u \in (0, 1].$$

This is equivalent to

$$\int_0^u f_{U^{(n)}}(u) - f_{\tilde{U}_{n,m}}(u)du.$$

This in turn is equivalent to

$$\int_0^u f_{U^{(n)}}(u)(1 - \frac{f_{\tilde{U}_{n,m}}(u)}{f_{U^{(n)}}(u)})du.$$

Then

$$F_{U^{(n)}}(u) - F_{\tilde{U}_{n,m}}(u) \leq \int_0^u f_{U^{(n)}}(u) \max_u (1 - \frac{f_{\tilde{U}_{n,m}}(u)}{f_{U^{(n)}}(u)}) du.$$

Note from Lemma 2.8.3,

$$\min_u \frac{f_{\tilde{U}_{n,m}}(u)}{f_{U^{(n)}}(u)} = \frac{f_{\tilde{U}_{n,m}}(1)}{f_{U^{(n)}}(1)} = \frac{m+1}{n}(1 - (\frac{m}{m+1})^n).$$

Then

$$F_{U^{(n)}}(u) - F_{\tilde{U}_{n,m}}(u) \leq F_{U^{(n)}}(u)(1 - \frac{m+1}{n}(1 - (\frac{m}{m+1})^n)).$$

Simplifying, one gets

$$F_{U^{(n)}}(u)(\frac{m+1}{n}(1 - (\frac{m}{m+1})^n)) \leq F_{\tilde{U}_{n,m}}(u).$$

Now from Lemma 2.8.3, it follows that $U^{(n)} \geq^{st} \tilde{U}_{n,m}$, combining this with the above equation we get

$$\frac{m+1}{n}(1 - (\frac{m}{m+1})^n) \leq \frac{F_{\tilde{U}_{n,m}}(u)}{F_{U^{(n)}}(u)} \leq 1.$$

Using the definition of $u$, and the fact that $F_{X^i}(\cdot)$ is an increasing function, the above equation can be rewritten as

$$\frac{m+1}{n}(1 - (\frac{m}{m+1})^n) \leq \frac{\Pr(F_{X^i}^{-1}(\tilde{U}_{n,m}) \leq r)}{\Pr(F_{X^i}^{-1}(U^{(n)}) \leq r)} \leq 1.$$

Note that $R_{mq}^i = F_{X^i}^{-1}(U^{(n)})$ and $\tilde{R}_{mq}^{i,m} = F_{X^i}^{-1}((\tilde{U}_{n,m})$, then the above equation can be written as

$$\frac{m+1}{n}(1 - (\frac{m}{m+1})^n) \leq \frac{\Pr(\tilde{R}_{mq}^{i,m} \leq r)}{\Pr(R_{mq}^i \leq r)} \leq 1.$$

To prove the second claim, recall that $G_{mq}^k(n) = \frac{E[F_{X^k}^{-1}(U^{(n)})]}{n}$. Note that $F_{X^k}^{-1}(\cdot)$ is an increasing function. Therefore it is sufficient to prove that $U^{(n)} \geq^{st} \tilde{U}_{n,m}$ to prove the theorem, which is shown to be true from Lemma 2.8.3.

We now prove the third part of the theorem. Note from the second part of the theorem, it is suffices to study

$$\frac{G^k_{mq}(n) - \tilde{G}^k_{mq}(n,m)}{G^k_{mq}(n)}.$$

Consider the difference between the two throughput, i.e., $E[F^{-1}_{X^k}(U^{(n)})] - E[F^{-1}_{X^k}(\tilde{U}_{n,m})]$. The difference can be expressed as

$$\int_0^1 F^{-1}_{X^k}(u) f_{U^{(n)}}(u) du - \int_0^1 F^{-1}_{X^k}(u) f_{\tilde{U}_{n,m}}(u) du.$$

Then following the methodology used in the first part of the proof on can show

$$E[F^{-1}_{X^k}(U^{(n)})] - E[F^{-1}_{X^k}(\tilde{U}_{n,m})] \leq \int_0^1 F^{-1}_{X^k}(u) f_{U^{(n)}}(u)(1 - \frac{m+1}{n}(1 - (\frac{m}{m+1})^n))du,$$

or

$$E[F^{-1}_{X^k}(U^{(n)})] - E[F^{-1}_{X^k}(\tilde{U}_{n,m})] \leq E[F^{-1}_{X^k}(U^{(n)})](1 - \frac{m+1}{n}(1 - (\frac{m}{m+1})^n)).$$

This completes the proof. □

# Chapter 3

# Providing Quality of Service while Exploiting Opportunism

## 3.1 Introduction

***Motivation.*** Data services required by users are a mixture of real-time streams (e.g., video/voice and multimedia) and best effort data transfers (like file downloads or web browsing). From a user's perspective this requires a scheduling scheme which can ensure quality of service (QoS) to a real-time session and/or minimize transfer delays associated with best effort sessions (see Figure 3.1). From a system perspective one would like the capability to admit a large number of real-time sessions, while at the same time, maximizing revenue generating data throughput. To manage such traffic mixes on limited wireless resources, one must be able to predict and evaluate the likelihood that QoS commitments can be met, i.e., devise complementary resource allocation and call admission strategies. Devising an opportunistic scheduling scheme that handles mixes of traffic while permitting some degree of performance prediction is the objective of this chapter.

***Challenges.*** As discussed in Chapter 1, capacity or data rate supported by wireless channel is not only time varying, but also heterogeneous across users. At the same time real-time and best effort sessions will have different traffic load statistics and heterogenous QoS requirements which a scheduler should somehow address. One might think of the base station scheduler as the meeting point where heterogeneity and variability in the wireless channels meets heterogeneity and variability in the requirements and offered traffic. Finding a practical approach to opportunistic scheduling that harmoniously

deals with these and at the same time enables the 'prediction' of performance towards supporting quality of service is the challenge we face.

***Related work.*** We discussed *proportionally fair* scheduling [13, 37] and the *exponential rule* [33] in Chapter 2. Additionally *modified-largest weighted delay first* [1] has been proposed. The algorithm as the name suggests, schedules the user currently having the highest product of user's weighted delay with its current channel condition. In [32] the performance of these three scheduling algorithms (along with maximum rate scheduling [16]) was compared from the perspective of providing QoS guarantees and the exponential rule was found to be best.

The above mentioned schemes try to provide QoS by attaching priority weights to users and choose to serve the user with the highest weighted channel capacity. The flexibility in assigning these weights allows one to handle heterogeneity in channel capacity distributions. It also allows a seamless integration of the real-time users with the best effort users while exploiting opportunism. However as we discussed in Chapter 2, these weights are complicated functions of the service a user has seen to date, the present queue backlog, and QoS or fairness requirements among users, etc., making proper selection of these weights difficult. As such it is unclear whether a meaningful performance prediction, resource management and call admission policies could be devised based on such schedulers.

The work of [40][41][39] explores realizing QoS guarantees based on an effective bandwidth concept. The approach largely focuses on the case where all users have homogeneous channel capacity distributions which is unlikely in practice. (The extension to the heterogeneous case is very unwieldy.) An evaluation of the offered QoS is based on determining the effective capacity which requires knowledge or estimation asymptotic log moment generating function of the channel capacity process seen by a user at the base station. Furthermore, because the underlying analysis is based on large buffer large

deviations, the resulting QoS estimate may not be useful on the short time scales relevant for real-time users. The shortcomings of this work highlights some of the difficulties we mentioned earlier. However, note that if we are to predictably ensure QoS guarantees it is likely that the knowledge of users' channel capacity statistics at the base station will be required.

There is very little work on opportunistic scheduling and the integration of real-time and best effort traffic. Due to the demanding nature of real-time users, a simple solution is give absolute priority to real-time over best effort traffic. If the real-time sessions were scheduled opportunistically then such scheme would enable one to exploit 'intra class opportunism', i.e., opportunism among users of the same class. Yet due to the coupling among real time streams, it is unclear how performance could be predicted. Furthermore, ideally one would like to also exploit 'inter class opportunism', i.e., opportunism from both the real-time and the best effort users competing for service.

***Contributions and organization.*** In this chapter we propose a novel opportunistic scheduling mechanism and resource allocation strategy that fulfil multiple objectives. Under the assumption that the (possibly heterogenous) channel capacity distributions of users are known (or estimated) at the base station and stay stable for moderate timescales, we are able to ensure probabilistic guarantees on the rate experienced by real-time sessions over short time scales. For simplicity, we begin in Section 3.2 by considering the case where sessions see independent, identically distributed channel capacity variations, i.e., homogeneous channel characteristics and want the same QoS guarantees. We develop stochastic lower bounds for the service received by real-time users which can be used as a basis for making admission control and resource allocation decisions. Then, in Section 3.3, we consider the case where users have heterogenous channel capacity variations and QoS requirements. Therefore the proposed opportunistic scheduler can *predictably guarantee QoS* over *short time scales* while still benefiting from opportunism

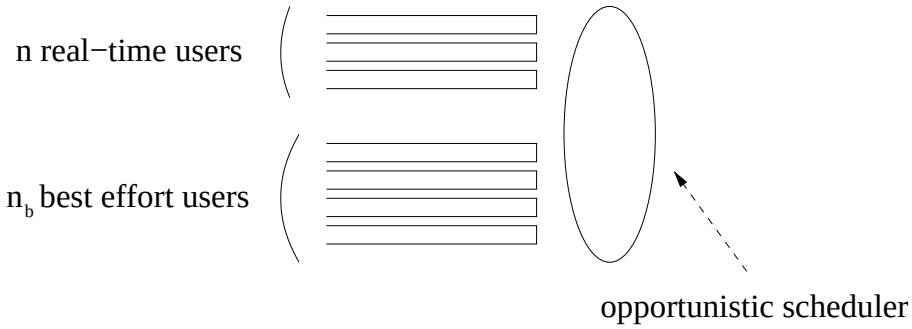



Figure 3.1: Scheduling a mixture of real-time and best effort users from a wireless base station.

when users have *heterogeneous channel capacity distributions* – i.e., exploiting both *intra class and inter class opportunism*. This is verified in the simulation results presented in Section 3.4 which show that we can satisfy strong QoS guarantees while achieving more than 90% of the system throughput realized under maximum rate scheduling. This is excellent since for a static saturated set of users, maximum rate maximizes the system throughput. Our analysis assumes users' channel capacities are fast fading, i.e., i.i.d., across slots, but we propose a heuristic modification that would make the scheduler robust if in fact users capacity variations were dominated by a slow fading environment. This claim is again supported by our simulations. Section 3.5 concludes the chapter.

## 3.2 Scheduling and Resource Allocation for Symmetrical Channel Capacity Distributions

### 3.2.1 System Model and Notation

For simplicity, we again only consider downlink scheduling from a base station to multiple users (the scheme can be applied for uplink scheduling as well). The system model is similar to that described in Chapter 2. Time is divided into equal sized slots with at most one user served per slot. During each slot, each user feeds back the channel capacity or rate it can support to the base station which in turn makes a decision on which user to serve.

***Channel assumptions.*** Ensuring QoS requires giving guarantees towards future service, this can be done only if the users' capacity distribution or some function of it is known or predicted at the base station. Also note that if a user's distribution changes, the guarantee given may or may not hold, thus one must constantly track and learn the distribution (or some function of it). If the channel is quasi-stationary on time scales where users' rate distribution estimates may be made reliably, then the base station can track and allocate resources as needed to ensure QoS goals are met. Of course, if users' capacity distributions are changing too fast, then it is virtually impossible to provide any kind of guarantee. In the sequel we shall make the following assumptions on the channel capacity seen by a user.

**Assumption 3.2.1.** *We assume the channel capacity (rate) for each user is a stationary ergodic process and these processes are independent, identically distributed (i.i.d.) across users. The channel capacity for each user is fast fading, i.e., the channel capacity for each user is independent across slots and remains constant during a slot. Further we assume that the marginal distribution for each user is either known a priori, or estimated by the base station.*

***Discussion of the assumptions.*** We have justified most of the assumptions in Chapter 2. However, note that here we need to know users' channel distribution for maximum quantile scheduling (which will be needed for asymmetrical channel capacity distributions), and to perform resource allocation. We have shown in Chapter 2 that the throughput penalty due to estimation errors in the users' channel distributions is not high for maximum quantile scheduling. Additionally as will be discussed in the final analysis, the base station only needs to know the mean and variance of a certain quantity users' channel distribution to perform resource allocation. Finally, the assumption that users' channels are identically distributed is made for sim-

plicity, this will be relaxed later when we incorporate heterogeneous channel capacities in our framework.

***Notation.*** We begin by introducing some notation relevant to this section. (A list of frequently used notation is given in Table 3.1 for subsequent reference.) For simplicity, the time period of a slot is fixed to a single time unit size. Let $X^i$ be a random variable representing the channel capacity distribution seen by User $i$ on a typical time slot. By Assumption 3.2.1, the channel is fast fading, therefore $X^i$ captures the rate distribution seen by User $i$ on any slot. Let $x^i(t)$ denote the realization of the channel capacity of User $i$ for time slot $t$. According to Assumption 3.2.1, the base station knows the distribution of the $X^i$ in addition to $x^i(t)$ for each user. Also since for now, we assume that the channel capacity distributions are i.i.d. across users , therefore we will sometimes drop the users' index in this section and denote $X^i$ by $X$, a random variable whose distribution is same as that of the channel capacity of any of the users in the system.

Let $A_r(t)$ denote the set of active real-time users at time slot $t$, i.e., if $i \in A_r(t)$ the base station will allow User $i$ to compete for the slot $t$. Note that the scheduling discipline will be responsible for deciding which real-time users are 'allowed' to contend for a slot. Also note that for convenience it is possible for an active real-time user in a slot to have no backlogged data. The set of active best effort users is denoted by $A_b(t)$. A best effort User $j$ is said to be active only if it has a backlog prior to that slot. The set of active best effort users is denoted by $A_b(t)$ and define $A(t) := A_r(t) \cup A_b(t)$.

Recall from Chapter 2 that under maximum rate scheduling, the base station receives channel capacity feedback $x^i(t)$ from each user in $A(t)$ and chooses the 'best' to serve. More formally, the access station chooses to serve User $i$ during slot $t$ if $x^i(t) = \max_{j \in A(t)} x^j(t)$. We let $X^{(l)} = \max\{X_1, ..., X_l\}$, where all $X_j$'s are i.i.d. and $X_j \sim X$, i.e., $X^{(l)}$ is the maximum of $l$ i.i.d. random variables. If User $i$ is selected on a slot when competing with $l-1$

other users under maximum rate scheduling, his conditional rate distribution $X^i$ is the same as $X^{(l)}$. This follows easily by symmetry among the contending users. Let us discuss some properties of $X^{(l)}$ which will be useful in the proofs given later. $X^{(l)}$ are stochastically increasing in $l$, i.e., $\forall x$, $\Pr(X^{(l+1)} \geq x) \geq \Pr(X^{(l)} \geq x)$, or $X^{(l+1)} \geq^{st} X^{(l)}$.

We shall let $n$ denote the total number of real-time users and $n_b$ the total number of best effort users (see Figure 3.1). For simplicity we assume that a user initiates only one type of session at a time, with exactly one real-time stream per real-time user, i.e., the number of real-time users is equal to the number of real-time streams.

***QoS definition.*** The notion of QoS considered in this chapter involves ensuring a User $i$ sees a desired rate $r$ over a frame of length $\tau$ with an outage probability of $\delta$. More formally, we divide time into equal sized 'frames' of $\tau$ units and our goal is to ensure that for each of these frames

$$\Pr(S_i(\tau) > r\tau) \geq 1 - \delta,$$

where $S_i(\tau)$ is a random variable denoting the cumulative potential service to User $i$ during a frame. For simplicity we restrict $\tau$ to take only integral values with respect to the time unit, i.e., the QoS guarantees are given only over an integral number of time slots.

If the traffic load of User $i$ does not exceed $r\tau$ over a given frame, and any data experiencing more than a delay of $2\tau$ is thrown away (i.e., no longer considered for scheduling), then the above rate guarantee translates to a delay guarantee of the form

$$\Pr(D^i \leq 2\tau) \geq 1 - \delta,$$

where $D^i$ is the scheduling delay associated with a typical bit of data designated for User $i$.

To guarantee the required QoS, we will use a stochastic envelope based approach [6][15]. The idea is to lower bound the actual service $S_i(\tau)$ by a

quantity $S_i^{low}(\tau)$ that satisfies two properties, firstly

$$S_i(\tau) \geq^{st} S_i^{low}(\tau),$$

so that if $S_i^{low}(\tau)$ meets the QoS guarantee then so will $S_i(\tau)$. Secondly, $S_i^{low}(\tau)$ will be analytically tractable from a resource allocation perspective.

We will first focus on providing the same QoS guarantee to all the real-time users, and later generalize to multiple QoS needs in Section 3.3.

### 3.2.2 Opportunistic Round Robin

Recall that our goal is to find a scheduling scheme and resource allocation strategy that exploits both intra and inter class opportunism to provide high throughput to all users while meeting real-time users' QoS requirements. Yet, let us first consider scheduling $n$ real-time users. A simple way to serve them is to use a frame with $n$ slots. In every slot, the users feedback their rate for that slot and the base station opportunistically serves the best user. Once a user has been served in a frame, he does not compete for service until the next frame. This ensures that each active real-time user gets served once every frame. This scheme is similar to that proposed in [14], however the objective there was not to provide QoS guarantees. One might call this 'opportunistic round robin' scheduling. Consider a fixed saturated system, under conventional round robin a typical user in such a system would see a slot whose rate distribution is the same as $X^{(1)}$, i.e., no multiuser diversity gain. However, under the opportunistic round robin scheme, a user is equally likely to be served on any one of the slots of the frame. If it is served on the $(n-j+1)^{th}$ slot, it would have competed with $j-1$ other users and will see a rate distribution of $X^{(j)}$. This means that the rate distribution in a *typical* slot will be a mixture, i.e., with probability (w.p.) $\frac{1}{n}$ it will see the distribution of $X^{(n)}$, w.p. $\frac{1}{n}$ a distribution of $X^{(n-1)}$ and so on. We let the random variable

$Y$ have the rate distribution seen by such a user, then

$$Y = \begin{cases} X^{(n)} \text{ w.p. } 1/n \\ \ldots \text{ w.p. } 1/n \\ X^{(1)} \text{ w.p. } 1/n. \end{cases} \tag{3.1}$$

Clearly $Y \geq^{st} X^{(1)}$, so our opportunistic round robin scheme will give improved data rate to users.

In present day systems, a time slot is of the order of milliseconds (1.67 msec for CDMA-HDR), while video and multimedia traffic require guarantees of around 100 kbps on a time scale of the order of hundreds of milliseconds. So, if the number of real-time users is not large, i.e., frame sizes are tens of milliseconds, there is a 'slack' in the QoS requirement that is not exploited by opportunistic round robin which in turn can lead to severe system throughput penalties–our simulations (not presented here) show this. This slack can be used to schedule best effort users and enhance opportunism. An alternative is to have a larger frame and give multiple slots to users. This brings us to our proposed scheduling scheme.

### 3.2.3 Proposed Scheduling Scheme

In our scheme the frame is as long as the time period on which the QoS guarantees need to be ensured, i.e., $\tau$. Each real-time user is assigned $k$ 'tokens', i.e., each real-time user will be served at most $k$ slots within a frame. Note that $nk$ can at most be equal to $\tau$. We describe how to determine the value of $k$ in the next subsection.

The proposed scheduling scheme combines a policy to decide which users will be active, i.e., the set $A(t)$ that contend for a slot, with a mechanism to select the user to serve during that slot. To avoid confusion, we henceforth refer to the latter as the 'selection criterion' and denote it by a set-valued function $\phi(\cdot)$. In this section we use maximum rate selection criterion among users, i.e.,

$$\phi(B(t)) := \arg\max_{j \in B(t)} x^j(t),$$

where $B(t)$ is a set of users at time slot $t$. Note since rate distributions are i.i.d., i.e., symmetric, this criterion is fair and maximizes system throughput.

We present the proposed scheduling scheme in terms of an algorithm that is implemented every frame. It starts at the first slot of the frame and ends at the last slot of the frame. The time slots within a frame are indexed as $t = 1, \ldots, \tau$, while the number of tokens remaining for User $j$ is denoted by $k_r^j$. Recall that $A_r(t)$ is the set of active real-time users allowed by the proposed scheduling scheme to compete during slot $t$, in contrast $A_b(t)$ is the set of best effort users that have data backlogged during slot $t$ and $A(t) = A_r(t) \cup A_b(t)$.

---

Algorithm for the proposed scheduling scheme

---

1. Initialize $t = 1$ and $A_r(1)$ to be the set of all admitted real-time users each with $k$ tokens allocated to it, i.e., $\forall j \in A_r(1)$, $k_r^j = k$.

2. If $t > \tau$, i.e., end of frame is reached, then go to Step 12, else if $(\tau - t) = \sum_{j \in A_r(t)} k_r^j$, i.e., the number of remaining slots in the frame is equal to the total number of remaining tokens, then go to Step 8 else go to the next step.

**Phase I**

3. Based on the feedback from the users, choose User $i$ such that $i \in \phi(A(t))$, with ties broken randomly.

4. If $i$ is a best effort user, then serve him and go to Step 7, else go to the next step.

5. If $i$ is a real-time user which is backlogged, then serve him, else if $A_b(t)$ is not empty serve a best effort user from $A_b(t)$. The best effort user can be selected using any criterion e.g. proportionally fair, maximum rate etc.

6. Update $k_r^i = k_r^i - 1$, and if $k_r^i = 0$, i.e., User $i$ has used up its tokens, then update $A_r(t) = A_r(t) \setminus \{i\}$, i.e., remove User $i$ from $A_r(t)$.

7. Define $A_r(t+1) = A_r(t)$ and increment $t = t + 1$. Go to Step 2.

**Phase II**

8. Based on the feedback, choose User $i$ such that $i \in \phi(A_r(t))$, with ties broken randomly. Note that we are now choosing only among real-time users.

9. If $i$ is a backlogged real-time user, then serve him, else if $A_b(t)$ is not an empty set, then serve a best effort user from $A_b(t)$. Again, the best effort user can be selected using any criterion e.g. proportionally fair, maximum rate etc.

10. Update $k_r^i = k_r^i - 1$, if $k_r^i = 0$, i.e., User $i$ has used all of his tokens, then update $A_r(t) = A_r(t) \setminus \{i\}$, i.e., remove User $i$ from $A_r(t)$.

11. Define $A_r(t+1) = A_r(t)$ and increment $t = t + 1$, if $t > \tau$, then go to the next step, else go to Step 8.

12. Proceed to the next frame.

---

We now give a brief description of the scheme using an example containing a number of best effort sessions and 2 real-time users. Each real-time user is assigned 3 tokens. Figure 3.2 shows a frame of size $\tau = 10$.

Whenever a real-time user is given a chance to be served, his token count decreases by 1 and when the token count becomes zero, he is no longer considered for service (Steps 6 and 10).

The scheduling scheme is divided into two phases. During the first phase (Steps 3 - 7), both active real-time and best effort users are allowed to compete for service. In each slot, the user with the maximum rate is identified and served, with ties broken randomly. The first phase continues until the *total number of remaining tokens in the system is equal to the number of slots remaining in the frame* (Step 2). In our example (Figure 3.2), the first phase lasts until Slot 7. In Slot 3 and 5, real-time User 1 supported the highest data rate and was served, similarly real-time User 2 was served during Slot 6. Best effort users were served in the rest of the slots (the shaded ones). After Slot 7, the above mentioned condition for the end of first phase is satisfied, so the second phase starts.

During the second phase (Steps 8 - 11), only the real-time users are allowed to contend for service under the maximum rate selection criterion. This phase is needed to ensure that every real-time user is served as many times as the number of tokens assigned to it. Note that $A_r(t)$ will be empty by the end of the frame. The second phase slot assignment for the example are shown in Figure 3.2. Slots 8 and 9 are assigned to real-time User 2 and he is no longer allowed to compete. As a result, User 1 gets Slot 10, thus ensuring that both users got served as many times as the number of tokens they were allocated. Note that Figure 3.2 is just one of the many realizations that the proposed scheduling scheme could follow (even the starting point of the second phase is not fixed). In fact the number of possible realizations grows combinatorially in both $n$ and $k$, making the scheme hard to analyze.

Note that we provision tokens for real-time users based on their QoS requirements, therefore it is possible that a real-time user selected for service may have no data to receive during that slot (note that the definition of an

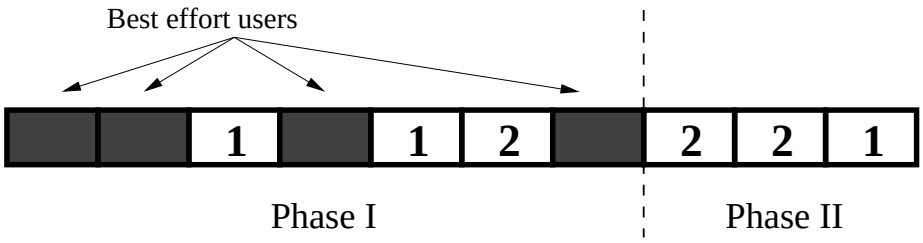



Figure 3.2: Example of the proposed scheduling scheme with frame size of 10 and 2 real-time users each having 3 tokens.

active real-time user allows this). When this is the case, we allow the slot to be used by any active best effort user (Steps 5 and 9). The best effort user to be served can be selected on a desirable criterion e.g. maximum rate, proportional fairness.

Some comments on the two phases of the proposed scheduling scheme. The first phase allows the exploitation of both inter and intra class multiuser diversity and thus takes advantage of the slack in the QoS requirement. The second phase is needed to guarantee quality of service to real-time users, however note that opportunism is still exploited across the remaining real-time users. Together the two phases allow one to maintain high throughput while providing quality of service.

### 3.2.4 Analysis and Resource Allocation

The value of $k$ must be decided so that the specified QoS guarantee is met for all real-time users. Let $Z_j^*$ denote the data sent to a real-time user upon consuming its $j^{th}$ token, i.e., the $j^{th}$ time it gets served. Our goal is to determine the minimum number of tokens $k$ such that $\Pr(\sum_{j=1}^{k} Z_j^* \geq r\tau) \geq 1-\delta$. This is not easy to compute, even when users have i.i.d. rate distributions.

Note that $\forall j,\ Z_j^* \geq^{st} X$, i.e., at worst a user contends with no other users and thus sees the marginal channel capacity distribution of a typical slot with no opportunistic gain. Therefore it is likely that $\forall s,\ \sum_{j=1}^{s} Z_j^* \geq^{st}$

$\sum_{j=1}^{s} X_j$, where $X_j$'s are i.i.d. and $X_j \sim X$. (Note that because $Z_j^*$'s are not independent random variables $Z_j^* \geq^{st} X$ is not a sufficient condition for proving $\sum_{j=1}^{s} Z_j^* \geq^{st} \sum_{j=1}^{s} X_j$, but the bound will be shown to be true later.) Then perhaps, the simplest bound would be to replace $Z_j^*$ by $X$, and finding the minimum value of $k$ that satisfies $\Pr(\sum_{j=1}^{k} X_j \geq r\tau) \geq 1-\delta$ using e.g., the Central Limit Theorem. But this bound is very conservative, i.e., will allocate too many tokens, because $X$ does not reflect any of the opportunistic gains achieved by the proposed scheduling scheme.

To find a more efficient, yet conservative resource allocation approach, consider a 'static division scheduling scheme', where the frame is divided into two parts. During the first part, consisting of $\tau - nk$ slots, the slots are opportunistically allocated among the best effort users, while the real-time users are opportunistically served during the second part. This is a special case of the original proposed scheduling scheme where only best effort users are served in Phase I. Let $Z_j$ be the same quantity for the static division scheme as $Z_j^*$ is for the proposed scheme. We claim in Theorem 3.2.2 that $\sum_{j=1}^{k} Z_j^* \geq^{st} \sum_{j=1}^{k} Z_j$, i.e., the static division scheduling scheme under performs relative to our proposed mechanism.

Before proving this claim, we digress to state three properties satisfied by both the proposed and the static division scheduling scheme when maximum rate selection is used under Assumption 3.2.1. These properties are used in the proof of Theorem 3.2.2, its supporting lemmas, and subsequent results.

*Property* 3.2.1. (*Equal Resource Allocation*) All real-time users are allocated an equal number $k$ of tokens.

*Property* 3.2.2. (*Symmetric Selection*) In a typical slot, each active real-time user is equally likely to be selected for service by the selection criterion (the selection probability for an active best effort user can be different).

*Property* 3.2.3. (*Monotonicity*) The selection criterion is such that for any User $i$ and for any value of $l$, $X^{i,(l+1)} \geq^{st} X^{i,(l)}$, where $X^{i,(l)}$ is the random variable denoting the rate seen by User $i$ given it is selected for service while competing with $l-1$ other users.

We now introduce some further notation. Let $N_j^*$ be a random variable representing the number of real-time users in the system when a typical real-time user gets the $j^{th}$ token in our proposed scheduling scheme. Let $N_j$ represent the same quantity for the static division scheduling scheme. Note that under the static division scheduling scheme, a real-time user competes only with other real-time users while under the proposed scheduling scheme there might also be competing best effort users. Therefore it is likely that $N_j^* \geq^{st} N_j$, this is at the root of our next theorem, which is proven in Appendix 3.6.1.

**Theorem 3.2.2.** *Consider the proposed and the static division scheduling schemes where all real-time users are allocated an equal number $k$ of tokens. Then under Assumption 3.2.1 and the maximum rate selection criterion, for a typical real-time user*

$$\sum_{j=1}^{k} Z_j^* \geq^{st} \sum_{j=1}^{k} Z_j.$$

Theorem 3.2.2 implies that to meet the quality of service constraint, it is sufficient to satisfy $\Pr(\sum_{j=1}^{k} Z_j \geq r\tau) \geq 1-\delta$. To compute this, let us study the properties of $Z_j$. Note that $Z_j$ is the maximum over $N_j$ i.i.d. random variables with the same distribution as $X$. In other words, $Z_j \sim X^{(l)}$ *w.p.* $\Pr(N_j = l)$, $\forall l$. The distribution of $X$ is assumed to be known, but it is difficult to calculate the distribution of $N_j$ because of the number of ways the static scheduling scheme can proceed, i.e., how users are served, grows in a combinatorial fashion. Also note that $Z_j$'s are not i.i.d. which makes it difficult to calculate $\Pr(\sum_{j=1}^{s} Z_j \geq r\tau)$ for any given value of $s$. To remedy this, we

propose a further stochastic lower bound that still factors the opportunistic gain. Our next claim is that $\sum_{j=1}^{k} Z_j \geq^{st} \sum_{j=1}^{k} Y_j$, where $Y_j$'s are i.i.d. and $Y_j \sim Y$, where $Y$ is as defined in (3.1) in Section 3.2.2. We shall refer to this stochastic lower bound as the 'mixture bound'. The following theorem formally states our claim with the proof given in Appendix 3.6.2.

**Theorem 3.2.3.** *Consider the static division scheduling scheme where all real-time users are allocated an equal number of $k$ tokens. Then under Assumption 3.2.1 and maximum rate selection criterion, for a typical real-time user*

$$\sum_{j=1}^{k} Z_j \geq^{st} \sum_{j=1}^{k} Y_j,$$

*where $Y_j$'s are i.i.d. and $Y_j \sim Y$, with $Y$ is as defined in (3.1).*

Theorem 3.2.3 gives a bound on the cumulative data received by a typical real-time user in a frame by a sum of i.i.d. random variables where each is a mixture of distributions. If the number of tokens required per user, i.e., $k$, is large enough, the distribution of $\sum_{j=1}^{k} Y_j$ can be roughly approximated, e.g. using the Central Limit Theorem. An advantage of using the Central Limit Theorem is that one can compute the value of $k$ based only on the mean and variance of $Y$, which eliminates the need to know the actual distribution of $X$. Of course note that if users' rate distributions change, then so will the number of tokens required by them and the value of $k$ will have to be recomputed and allocated to track such changes.

Note that by virtue of the definition of $Y$, the above approach factors the opportunistic gains in our scheme. Recall that the simplest bound to compute $k$ conservatively would be to ensure $\Pr(\sum_{j=1}^{k} X_j \geq r\tau) \geq 1-\delta$. Now as discussed in Subsection 3.2.2 $Y \geq^{st} X$, and due to independence among $Y_j$'s, $\sum_{j=1}^{k} Y_j \geq^{st} \sum_{j=1}^{k} X_j$. Hence once can conclude that $\sum_{j=1}^{k} Z_j^* \geq^{st} \sum_{j=1}^{k} X_j$, i.e., the simplest method is conservative. Figure 3.3 summarizes the overall stochastic ordering for the cumulative data received by a typical real-time user

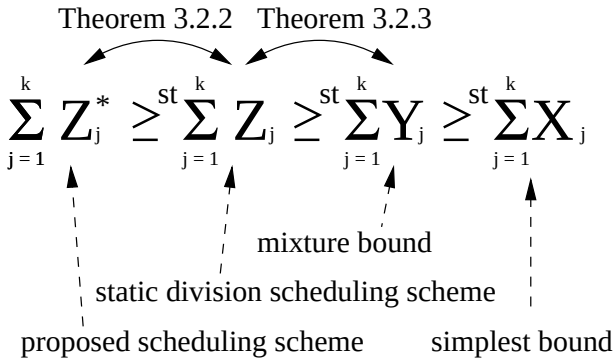



Figure 3.3: Stochastic ordering of the cumulative data received by a typical real-time user in a frame under the proposed scheduling scheme, the static division scheduling scheme, the mixture method and the simplest method.

under the proposed scheduling scheme, the static division scheduling scheme, the mixture bound and the simplest bound.

***A simple numerical experiment.*** As mentioned earlier, the simplest bound may allocate too many tokens. For example, we computed the number of required tokens per user using both the simplest bound and the proposed mixture bound for a system where each real-time user required a rate guarantee of 100 kbps over a time scale of 167 msec with an outage of 1%. The number of real-time users was 5 and all users were experiencing Rayleigh fading with a mean signal to noise ratio(SNR) of 2. Each slot was of size 1.67 msec (so the frame size was 100 slots) and the mapping from SNR to discrete rates was that used for CDMA-HDR [2]. The simple bound gave a requirement of 20 tokens per user while the mixture bound suggested only 12 tokens were needed. In addition, simulations showed that the exact number of tokens required to meet the guarantee were 11. *This suggests that our mixture bound is fairly tight, and thus useful.*

We emphasize that under the proposed scheme, unlike the weight based schemes discussed in related work, we were able to develop a concrete resource allocation approach.

## 3.3 Scheduling and Resource Allocation for Asymmetrical Channel Capacity Distributions

The symmetrical rate distributions case considered above, though unrealistic is a good starting point to solving the more general problem. In this section we generalize to the case where users experience different channel capacity distributions and describe the modifications required to our proposed scheme. We restate our assumption on the users' channel characteristics as follows:

**Assumption 3.3.1.** *We assume the channel capacity (rate) for each user is a stationary ergodic process and these processes are independent, but* not *necessarily identically distributed across users. The channel capacity for each user is fast fading, i.e., the channel capacity for each user is independent across slots and remains constant during a slot. Further we assume that the marginal distribution for each user is either known a priori, or estimated by the base station.*

Note that we are not assuming any specific distribution on the channel capacity variation.

The token scheme proposed in Section 3.2 achieves multiple goals, it guarantees that the QoS requirements for real-time users are met, while exploiting both intra and inter class opportunism to achieve high overall throughput. We want these desirable properties to hold while extending the scheme to the asymmetric case. Our approach of allocating tokens to each real-time user and then scheduling users opportunistically allows us to achieve these goals. However to efficiently calculate the number of tokens required by a user, one would like the Theorem 3.2.2 and Theorem 3.2.3 to also hold under Assumption 3.3.1. As mentioned earlier, the proofs of these theorems depend on the Properties 3.2.1, 3.2.2 and 3.2.3 holding true.

Let us consider the 'Equal Resource Allocation' property. Under Assumption 3.3.1, it is likely that different users may require different number of tokens to be guaranteed the same QoS. This can be dealt with by simply over allocating tokens so that all real-time users have the same number of tokens, but this in turn can lead to lesser number of real-time users getting admitted. Better alternatives will be discussed later.

The 'Symmetric Selection' and 'Monotonicity' properties depend on the selection criterion. It is clear that under Assumption 3.3.1, it is unlikely that maximum rate selection criterion will satisfy Property 3.2.2. An alternative is to randomly select a user (among the active ones), however there would be no opportunistic gains in this case. Our solution is to use maximum quantile scheduling, which will ensure that the two properties are satisfied and yet give good opportunistic gains.

From [22][23] one knows that maximum quantile scheduling satisfies Symmetric Selection, i.e., each competing user is *equally likely* to get served on a typical slot. It is easy to show that maximum quantile scheduling satisfies Property 3.2.3, i.e., Monotonicity. Define $X^{i,(l)} = \max\{X_1^i, \ldots, X_l^i\}$, where $X_j^i$'s are i.i.d. and $X_j^i \sim X^i$. Then the rate experienced by User $i$ when selected for service on a typical slot by maximum quantile scheduling while competing with $l-1$ other users, has the same distribution as $X^{i,(l)}$. It is easy to see that for any $l$, $X^{i,(l+1)} \geq^{st} X^{i,(l)}$, i.e., Monotonicity is satisfied.

We are now in a position to describe the proposed modification to our scheduling discipline under Assumption 3.3.1.

### 3.3.1 Proposed Modification

We begin by discussing resource allocation, i.e., evaluating how many tokens should be allocated to each user. In order to do so, we define a new quantity $Y^i$ given by

$$Y^i = \begin{cases} X^{i,(n)} \text{ w.p. } 1/n \\ \dots \quad \text{w.p. } 1/n \\ X^{i,(1)} \text{ w.p. } 1/n. \end{cases} \tag{3.2}$$

As mentioned earlier, it is likely that due to the asymmetric nature of users rate distributions, each real-time user may require a different number of tokens for the same QoS requirement. For each real-time user, calculate

$$k^i = \min_s \{s \mid \Pr(\sum_{j=1}^{s} Y_j^i \geq r\tau) \geq 1 - \delta\}, \tag{3.3}$$

where $Y_j^i$ are i.i.d. and $Y_j^i \sim Y^i$, with $Y^i$ defined in (3.2). We shall let $k$ now be given by

$$k = \max_{j=1,\dots,n} k^j. \tag{3.4}$$

Suppose every real-time user is allocated $k$ tokens. Note that we require that, $nk \leq \tau$, i.e., the total number of tokens allocated must be less than or equal to the size of frame. Also note that even though taking the maximum over $k^j$ seems to be a restrictive, we will discuss ways to overcome this requirement later in this subsection.

It should be clear by now that the selection criterion is changed to maximum quantile instead of maximum rate. Thus the selection criterion in the algorithm is now defined as

$$\phi(B(t)) := \arg\max_{j \in B(t)} F_j(x^j(t)),$$

when the $X^j$ are continuous. The suitable modification for the discrete case is described in Chapter 2.

With the proposed modifications, the three properties stated in the previous section are satisfied. It follows that the claims of Theorem 3.2.2 and 3.2.3 hold under Assumption 3.3.1. This in turn shows that the value of $k$ obtained in (3.3) and (3.4) will be conservative.

Rather than state the modified versions of Theorem 3.2.2 and 3.2.3 under Assumption 3.3.1, we will state a stronger version that will be useful

later in the sequel. Let $S$ be any set such that $S \subseteq \{1, \ldots, k\}$, this can be viewed as any subset of the tokens assigned to a user. Let $Z_j^{i*}$ denote the transmitted data to real-time User $i$ when it uses up the $j^{th}$ token under the proposed scheduling scheme and let $Z_j^i$ be the same quantity for the static division scheduling scheme. The following are the generalized theorem statements without proofs (which are analogous to those of Theorem 3.2.2 and 3.2.3).

**Theorem 3.3.2.** *Consider the proposed and the static division scheduling schemes where all real-time users are allocated an equal number $k$ of tokens. Then under Assumption 3.3.1 and maximum quantile selection criterion, for any real-time User $i$*

$$\sum_{j \in S} Z_j^{i*} \geq^{st} \sum_{j \in S} Z_j^i,$$

*for $S \subseteq \{1, \ldots, k\}$.*

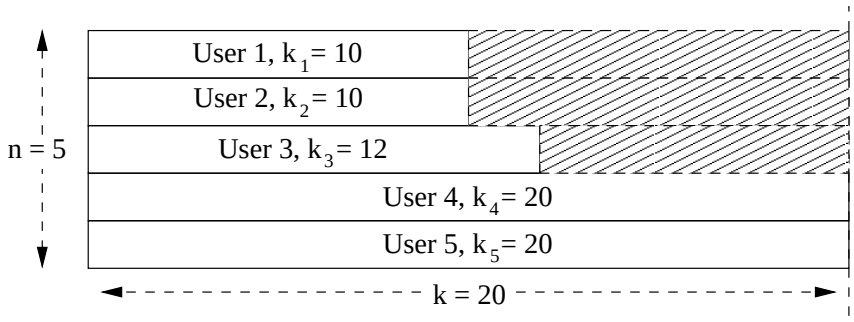
**Theorem 3.3.3.** *Consider the static division scheduling scheme where all real-time users are allocated an equal number $k$ of tokens. Then under Assumption 3.3.1 and maximum quantile selection criterion, for any real-time User $i$*

$$\sum_{j \in S} Z_j^i \geq^{st} \sum_{j \in S} Y_j^i,$$

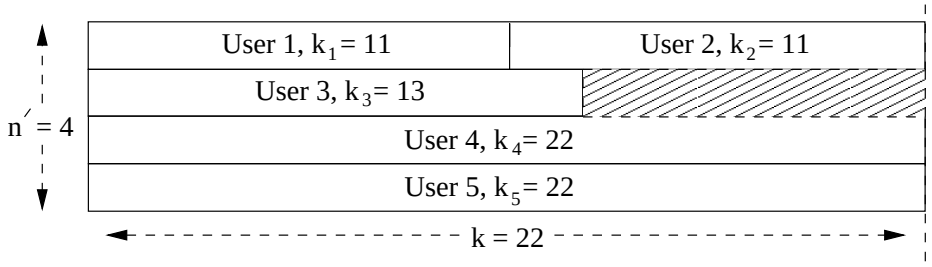
*for $S \subseteq \{1, \ldots, k\}$, where $Y_j^i$'s are i.i.d. with the same distribution as $Y^i$ is given by (3.2).*

***Grouping of users.*** As mentioned earlier, allocating the same number of tokens $k$ based on (3.3) and (3.4) is likely to be conservative for heterogeneous users. To improve upon this, we group users with smaller token requirements into single virtual users. We explain this with an example below.

Consider the following scenario, suppose there are 5 real-time users in the system. All users undergo Rayleigh fading, but have different mean SNR. User 1 and 2 have a mean SNR of 3, User 3 has a mean SNR of 2, while User




(a) no grouping




(b) User 1 and 2 grouped together

Figure 3.4: Parts (a) and (b) show token requirements and allocation with and without grouping respectively. The shaded portion depicts the excess allocated tokens.

4 and 5 have a mean SNR of 0.8. The SNR to rate mapping is the same as the example discussed in Section 3.2.4, i.e., same as that of CDMA-HDR. All real-time users are to meet a QoS guarantee of 100 kbps over a time scale of 167 msec with an outage probability of 1%. The frame size is thus 100 slots.

If tokens are allocated according to (3.3) and (3.4), then each real-time user would be allocated 20 tokens each (see Figure 3.4(a)), and there would be no slots left for Phase I of the proposed scheduling scheme. However, if a given real-time user competes with at most 3 other real-time users in a slot, then $Y^i = X^{i,(l)}$ $w.p.$ $\frac{1}{4}$, $l = 1, \ldots, 4$. In this case Users 1 and 2 will require 11 tokens each, while User 3 requires 13 tokens and User 4 and 5 require 22 tokens each. One can then allocate 22 tokens to User 3, 4 and 5 and combine User 1 and 2 into a single virtual user having a total of 22 tokens. This is illustrated in Figure 3.4(b), there $n'$ represents the maximum number of real-time users

that are allowed to compete for a slot (note that $n' = n$ if no grouping is used, else $n' < n$). As shown, User 1 uses the first 11 tokens of the virtual user followed by User 2. Then $S = \{1, \ldots, 11\}$ for User 1 and by Theorem 3.3.2 and 3.3.3, both would be able to meet their QoS requirement. We simulated such a system with 16 best effort users and verified our claim to be true.

The advantage of grouping is exhibited in this example where instead of 100 tokens, only 88 need to be allocated to real-time users.

There are multiple ways of grouping users together, one can also group more than two users. Another possibility is to increase $k$ slightly to allow better groupings. Referring back again to our example, suppose User 3 and 4 required 21 tokens each instead of the 22 required (with grouping), then one could have defined $k$ as 22, i.e., over allocate by 1 token to User 3 and 4, to allow us to group User 1 and 2.

Unfortunately finding the optimal grouping is a NP-Hard problem. We introduce some notation to prove our claim. Let $\mathcal{P}_{n'}$ denote the collection of all partitions of the set of all real-time users into $n'$ non-empty sets. Let $P$ denote a partition of the set of all real-time users, and $p$ be a set in $P$. We denote the tokens required by User $i$ when it is competing for service with $n'$ virtual real-time users for service by $k^i(n')$. Then the problem of optimal grouping can be written as follows:

*Optimal grouping problem*: Find the number of groups $n'$ and a partition $P$ of all real-time users into that number of groups such that

$$\min_{n'=1,\ldots,n} n' k_{max}(n'),$$

where

$$k_{max}(n') = \min_{P \in \mathcal{P}} \max_{p \in P} \sum_{i \in p} k^i(n').$$

The following theorem shows that the above defined problem of optimal grouping is NP-Hard.

**Theorem 3.3.4.** *The Optimal grouping problem is NP-Hard.*

*Proof.* Consider a fixed $n'$, then finding the value of $k_{max}(n')$ is equivalent to the load balancing problem, which in turn is known to be NP-Hard [10]. □

One can however propose simple heuristics to find suboptimal grouping solutions. For example consider a given $n'$, then a user must belong to one of the $n'$ groups, each corresponding to a single/virtual user. A simple solution would be to order users by their 'load' $k^i(n')$ and starting with the highest $k^i(n')$, place them in a group that currently has the lowest total load. One can search over different best fit solutions varying values of $n'$ and find the best solution. For other heuristics, see [10][8].

***Multiple QoS Guarantees.*** Let us consider providing different rate guarantees to different users. Here, each user can ask for a specific rate guarantee $r^i$ with his own outage probability $\delta^i$. However, the time scale over which the guarantee is given, i.e., the frame length $\tau$ is common to all users. Supporting multiple QoS requirements can lead to different users needing different numbers of tokens, which can be solved by grouping real-time users together. Thus extending our scheme to meet multiple QoS criteria efficiently.

### 3.3.2 Call Admission Policy

The call admission policy is quite simple, to admit a call $n'k \leq \tau$, where $k$ now is the number of tokens allocated to each user or a virtual user (if there is grouping).

However, note that in order to check whether a new user can be admitted into the system we have assumed that the capacity distribution of the new user $F_{n+1}(\cdot)$ is known a priori, this is unlikely. A practical solution to this problem is to instead use a typical distribution derived from users currently

or previously associated with the base station. For example, let $\tilde{F}(\cdot)$ be the ensemble average of the distributions for ongoing (or past) users, e.g.,

$$\tilde{F}(x) = \frac{\sum_{j=1}^{n} F_j(x)}{n}.$$

This distribution represents what a typical user might see, or what a mobile user might see throughout its lifetime in the system. In a practical setting $F_{n+1}(\cdot)$ could be initially set to $\tilde{F}(\cdot)$ until the actual distribution of the user has been tracked and estimated.

It is also important to note here that call admission is a long term decision, and one may need to save resources for future events like time varying rate distributions. Here, the number of tokens required by a user may vary across frames, this can be due to inaccuracy in estimating the distribution of users (especially for the newly admitted user) and time varying nature of users' rate distributions. Therefore one needs to reserve a pool of extra slots to handle such variations and allocate tokens from the pool to users that are not able to meet their QoS requirement in a frame. This pool can also be used for incoming handoffs from neighboring cells. Estimating the number of tokens that need to be reserved is an interesting question, and can be investigated as future work.

## 3.4 Simulation Results

We simulated the proposed scheme under various scenarios. We begin by considering the overall throughput performance as the number of real-time users and the QoS constraint vary. Next we observe the outage of real-time users with an increasing number of best effort users. Finally we propose a heuristic to accommodate slow fading channels and observe its performance.

Our simulation setup is similar to that of CDMA-HDR, the slot time period was set to 1.67 ms, with SNR to rate mapping borrowed from [2]. All

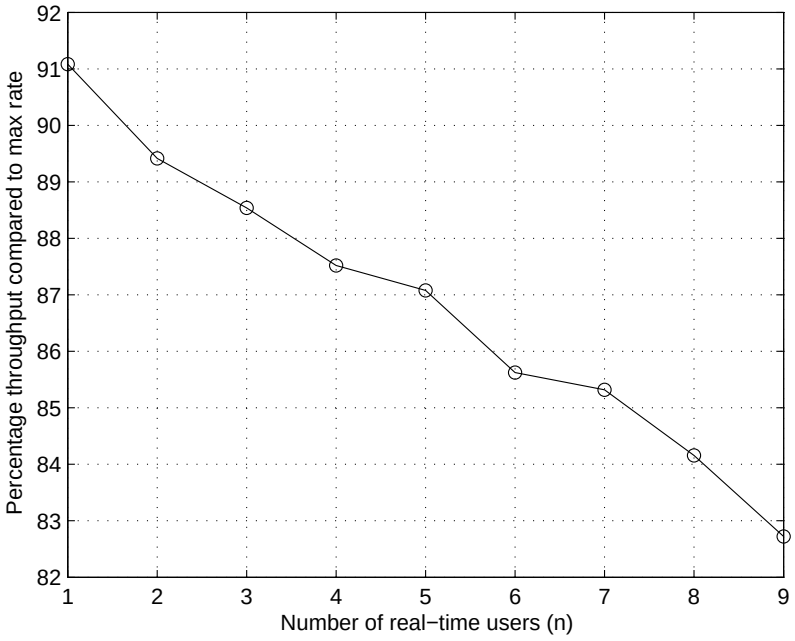



Figure 3.5: Percentage system throughput achieved by the proposed scheduling scheme compared to maximum rate scheduling with increasing number of real-time users.

users have i.i.d. Rayleigh fading channel capacity distribution with a mean SNR of 2.

### 3.4.1 Throughput Performance

In the first simulation, we investigate the overall system throughput as the number of real-time users increases. Each real-time user requires a guarantee of 100 kbps over a time scale of 167 msec (100 slots) with an outage of 1%. The total number of users is fixed at 20, while the number of real-time users increases from 1 to 9. For a given number of real-time users, the number of tokens required by each user was calculated using the mixture bound and the system was simulated by allocating these resources to each real-time user. To put our throughput results in perspective, we theoretically calculated the overall system throughput that would be achieved by the 20 users under maximum rate scheduling with no QoS constraint. Here, we remind the reader

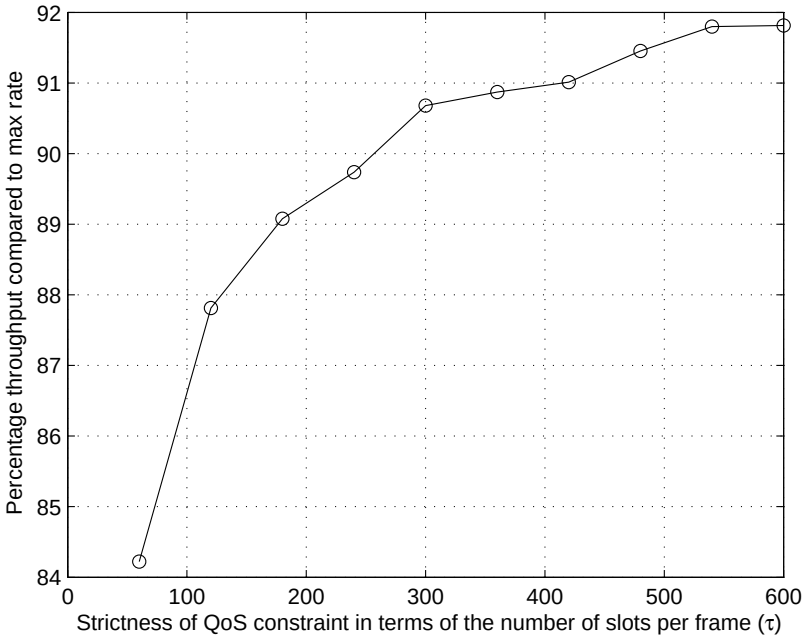



Figure 3.6: Percentage system throughput achieved by the proposed scheduling scheme compared to maximum rate scheduling with varying QoS constraint in number of slots per frame.

that maximum rate scheduling *maximizes overall system throughput* that can be achieved. Also note that keeping the total number of users constant allows us to compare the throughput achieved by our scheme to a constant quantity. The throughput achieved by our scheme as a percentage relative to this upper theoretical bound is are plotted in Figure 3.5. The first observation is that we are able to achieve more than 90% of throughput with 1 real-time user. Second, note that while the number of real-time users increases from 1 to 9, the throughput degradation experienced is less than 9%. This indicates that our scheme is quite robust to increases in the number of real-time users in terms of degradation in the overall system throughput.

In our second set of simulations, we studied the tradeoff in the overall system throughput as the QoS requirements were relaxed. The total number of users in the system was set to 20 with the number of real-time users fixed at 5. Each real-time user was given a guarantee of 100 kbps with an outage of

1% over varying frame sizes. The number of slots in a frame was varied from 60 (100 msec) to 600 (1 sec) in steps of 60. We again plotted the throughput achieved as a percentage of that possible under maximum rate. The results are shown in Figure 3.6. As expected, the system throughput grows as the QoS constraint is relaxed. However, note that the gains saturate quite quickly. One gets more than 6% gain in the first half of the plot, i.e., going from 60 slots per frame to 300 slots per frame, but only about 1% gain when going from 300 to 600 slots. We also observe that even for the most strict QoS requirement our scheme is able to achieve more than 84% of the capacity which like Figure 3.5 indicates that our scheme is very throughput efficient.

This simulation highlights the tradeoff between the strictness of the QoS constraint and the system throughput. If the QoS constraint is too tight, then it may not make sense to glean opportunistic gain by mixing the two types of users.

### 3.4.2 Outage versus Number of Best Effort Users

We now study the outage experienced by a real-time user as the number of best effort users increases. This is interesting because as the number of best effort users increase, real-time users are more likely to get served only during the second phase of the proposed scheduling scheme. Since the second phase is less throughput efficient than the first, the outage probability of a real-time user should increase with the number of best effort users. However since our bounds are calculated for the worst case scenario, each real-time user should still be able to meet his requirement. We note that the number of ongoing best effort sessions is likely to vary over the lifetime of a real-time session, and it is preferable not to require admission control on such sessions. Our simulations verify that the proposed scheduling scheme provides sufficient protection to real-time users even when the number of best effort users vary arbitrarily.

The setup was similar to the previous subsection. Each real-time user

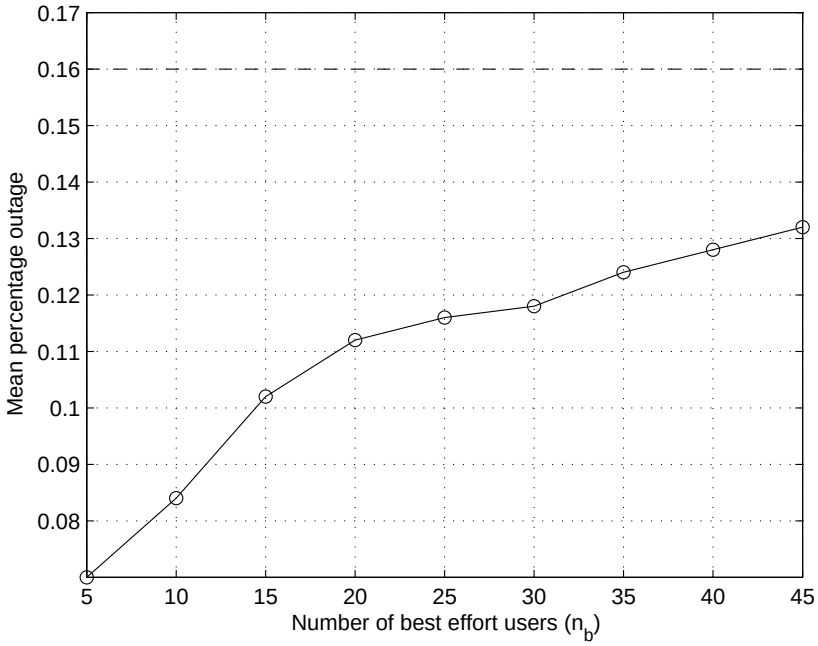



Figure 3.7: Mean percentage outage experienced by real-time users with increasing number of best effort users. The dotted line represents the outage obtained by just simulating the second phase of the proposed scheduling scheme.

required a guarantee of 100 kbps over a time scale of 167 msec (100 slots) with an outage of 1%. The number of real-time users was fixed to 5, while the number of best effort users was varied from 5 to 45 in steps of 5. We also simulated the worst case scenario, where all real-time users would be served only in the second phase. Figure 3.7 shows the results in terms of mean percentage outage. The dotted line corresponds to the outage in the worst case scenario. Observe that the mean outage is almost a magnitude lower than the guaranteed outage. This is because a lower bound is used in computing the required number of tokens and an integral number of tokens are allocated to a real-time user. Next observe that even when the number of best effort users is 45, i.e., 9 times the number of real-time user, there is still a reasonable gap in the outage when compared to the worst case scenario. This indicates that the real-time users get some gain from the first phase even for a large number of best effort users.

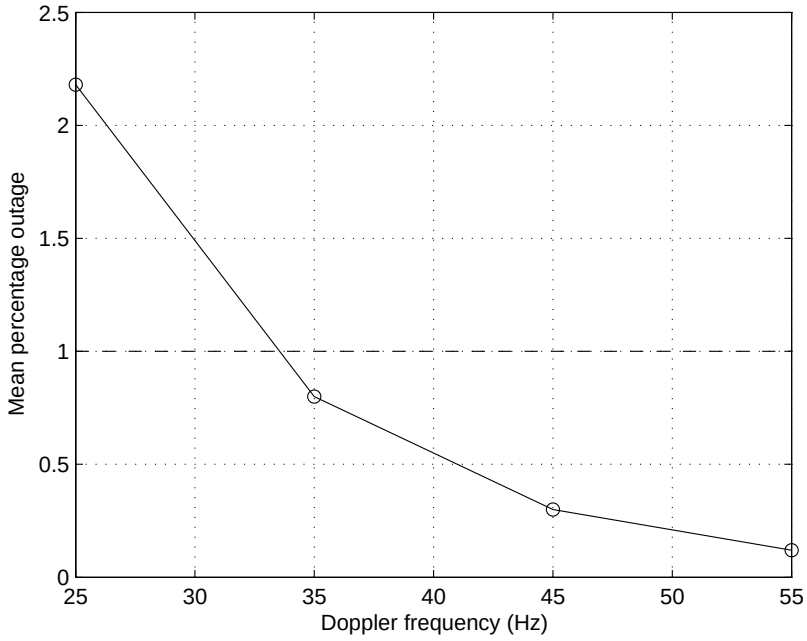



Figure 3.8: Mean percentage outage experienced by real-time users under token borrowing scheme and slow fading channels. The dotted line represents the target outage.

### 3.4.3 Outage Performance under Slow Fading

Our analysis in this chapter assumes fast fading channels, i.e., independent channel realizations per slot, however this may be optimistic. We now consider channels with slow fading characteristics. Our simulations show that the proposed scheme does not perform well in such channel conditions. We remedy this by proposing a heuristic.

In a frame some real-time users may be experiencing higher fades than their mean, while other real-time users might be suffering low fades. Then those experiencing high fades will require fewer allocated tokens and vice versa. This immediately suggests the possibility of token borrowing among users, i.e., users undergoing high fade allow other users to borrow some of their slots. If a real-time user satisfies his data requirement before finishing his allocated tokens, his remaining tokens are placed in a virtual pool. Whenever a real-time

user finishes his allocated quota of tokens without satisfying his requirement, he can borrow tokens from the virtual pool until his requirement is satisfied or the pool is exhausted.

We simulated the performance of the proposed heuristic under varying degree of correlation (in time) of users' channel. There were a total of 20 users in the system with 5 real-time users. Each real-time user is given a guarantee of 100 kbps over a time scale of 167 msec with an outage of 1%. The degree of correlation in a user's channel is varied by increasing the Doppler frequency associated with the channel from 25 Hz to 55 Hz in steps of 10 Hz. The mean percentage outage experienced across users is observed for each step are plotted in Figure 3.8. Observe that the proposed heuristic is able to meet its requirement for Doppler frequencies higher than that of 35 Hz.

## 3.5 Conclusion

In this chapter we proposed a scheduling and resource allocation scheme that allowed base station to serve a mixture of real-time and best effort users. The proposed scheme realizes probabilistic QoS guarantees over short time scales to real-time users while exploiting both intra and inter class opportunism across users. The effectiveness of the proposed approach is validated by simulation results. The proposed scheme also did away with the conventional approach of providing QoS by tuning relative weights among users. We also developed a simple call admission policy for the proposed scheme. A unique advantage of the proposed approach is that it supports users with arbitrary channel capacity distributions, this makes the scheme amenable to real world scenarios. Finally we proposed a heuristic for channels with slow fading characteristics.

## 3.6 Appendix

### 3.6.1 Proof of Theorem 3.2.2

Before presenting the proof, we introduce some notation. A vector of quantities say $W_j$ is represented as $\overrightarrow{W}_{1:l} = (W_1, \ldots, W_l)$. For any two vectors $\overrightarrow{W}_{1:l}$, $\overrightarrow{V}_{1:l}$, $\overrightarrow{W}_{1:l} \geq \overrightarrow{V}_{1:l}$ means that for all $j = 1, \ldots, l$, $W_j \geq V_j$. In other words, $\overrightarrow{W}_{1:l}$ is componentwise greater than $\overrightarrow{V}_{1:l}$. Recall that $N_j^*$ is the random variable representing the number of active real-time users in the system when a typical real-time user gets the $j^{th}$ token in the proposed scheduling scheme and $N_j$ be the same quantity for the static division scheduling scheme. Then $\overrightarrow{N}^*_{1:k}$ and $\overrightarrow{N}_{1:k}$ are the vector representation of $N_j^*$ and $N_j$ respectively. We begin by proving the following lemma.

**Lemma 3.6.1.** *Consider the proposed and static division scheduling scheme with all real-time users being allocated an equal number of k tokens. Then for a typical real-time user under Assumption 3.2.1 and maximum rate selection criterion*

$$\Pr(\overrightarrow{N}^*_{1:k} = \overrightarrow{n}_{1:k}) = \Pr(\overrightarrow{N}_{1:k} = \overrightarrow{n}_{1:k}) \tag{3.5}$$

*for any vector* $\overrightarrow{n}_{1:k}$.

*Proof.* For the proposed scheduling scheme, consider only those slots in which real-time users are served. There are exactly $nk$ slots of this type. If one considers the relative slot assignment possibilities among real-time users in these $nk$ slots, then the number of possible realizations is $\binom{nk}{k \ldots k}$.

Now consider a slot among these $nk$ slots, say the $l^{th}$ one. Then due to Property 3.2.2, every active real-time user during that slot is equally likely to get selected for service. Again due to Property 3.2.1 and 3.2.2, every real-time user is equally likely to be competing or active during the $l^{th}$ slot. Then if we average over all realizations of the proposed scheduling scheme, each user is equally likely to get assigned the $l^{th}$ slot.

Then the probability of a realization of the proposed scheduling scheme in terms of the assignment of the $nk$ slots among the real-time users is given by $\frac{1}{\binom{nk}{k\ldots k}}$. Consider realizations with $\overrightarrow{N}^*_{1:k} = \overrightarrow{n}_{1:k}$ for a particular real-time user. Let there be $h_{\overrightarrow{n}_{1:k}}$ such realizations, i.e, with $\overrightarrow{N}^*_{1:k} = \overrightarrow{n}_{1:k}$ for the user. Then

$$\Pr(\overrightarrow{N}^*_{1:k} = \overrightarrow{n}_{1:k}) = \frac{h_{\overrightarrow{n}_{1:k}}}{\binom{nk}{k\ldots k}}.$$

Similarly for the static division scheduling scheme,

$$\Pr(\overrightarrow{N}_{1:k} = \overrightarrow{n}_{1:k}) = \frac{h_{\overrightarrow{n}_{1:k}}}{\binom{nk}{k\ldots k}}.$$

Then clearly

$$\Pr(\overrightarrow{N}^*_{1:k} = \overrightarrow{n}_{1:k}) = \Pr(\overrightarrow{N}_{1:k} = \overrightarrow{n}_{1:k}).$$

□

Next we present the proof for Theorem 3.2.2.

*Proof.* Recall that $\sum_{j=1}^{k} Z^*_j \geq^{st} \sum_{j=1}^{k} Z_j$ means that for any $z$,

$$\Pr(\sum_{j=1}^{k} Z^*_j \geq z) \geq \Pr(\sum_{j=1}^{k} Z_j \geq z). \tag{3.6}$$

To prove this, we will show that for any vector $\overrightarrow{z}_{1:k} = (z_1, \ldots, z_k)$,

$$\Pr(\overrightarrow{Z}^*_{1:k} \geq \overrightarrow{z}_{1:k}) \geq \Pr(\overrightarrow{Z}_{1:k} \geq \overrightarrow{z}_{1:k}).$$

Conditioning on the number of real-time users present in each of the slots, we get

$$\begin{aligned}\sum_{\overrightarrow{n}_{1:k}} \Pr(\overrightarrow{Z}^*_{1:k} \geq \overrightarrow{z}_{1:k} | \overrightarrow{N}^*_{1:k} = \overrightarrow{n}_{1:k}) \Pr(\overrightarrow{N}^*_{1:k} = \overrightarrow{n}_{1:k}) \geq \\ \sum_{\overrightarrow{n}_{1:k}} \Pr(\overrightarrow{Z}_{1:k} \geq \overrightarrow{z}_{1:k} | \overrightarrow{N}_{1:k} = \overrightarrow{n}_{1:k}) \Pr(\overrightarrow{N}_{1:k} = \overrightarrow{n}_{1:k}).\end{aligned} \tag{3.7}$$

From Lemma 3.6.1, we know that

$$\Pr(\overrightarrow{N}^*_{1:k} = \overrightarrow{n}_{1:k}) = \Pr(\overrightarrow{N}_{1:k} = \overrightarrow{n}_{1:k}).$$

Then to prove (3.7), we need to show that

$$\Pr(\overrightarrow{Z}^*_{1:k} \geq \overrightarrow{z}_{1:k} | \overrightarrow{N}^*_{1:k} = \overrightarrow{n}_{1:k}) \geq \Pr(\overrightarrow{Z}_{1:k} \geq \overrightarrow{z}_{1:k} | \overrightarrow{N}_{1:k} = \overrightarrow{n}_{1:k}).$$

Let $M^*_j$ be the random variable representing the number of active best effort users when a typical real-time user gets selected the $j^{th}$ time. Then $\overrightarrow{M}^*_{1:k}$ is the vector representation of the $M^*_j$. Conditioning the left hand side of (3.8) on $M^*_j$, we get

$$\sum_{\overrightarrow{m}_{1:k}} \Pr(\overrightarrow{Z}^*_{1:k} \geq \overrightarrow{z}_{1:k} | \overrightarrow{N}^*_{1:k} = \overrightarrow{n}_{1:k}, \overrightarrow{M}^*_{1:k} = \overrightarrow{m}_{1:k}) \Pr(\overrightarrow{M}^*_{1:k} = \overrightarrow{m}_{1:k}) \geq \Pr(\overrightarrow{Z}_{1:k} \geq \overrightarrow{z}_{1:k} | \overrightarrow{N}_{1:k} = \overrightarrow{n}_{1:k}).$$

We also know that channel variations are independent across slots, thus we have that

$$\Pr(\overrightarrow{Z}^*_{1:k} \geq \overrightarrow{z}_{1:k} | \overrightarrow{N}^*_{1:k} = \overrightarrow{n}_{1:k}, \overrightarrow{M}^*_{1:k} = \overrightarrow{m}_{1:k}) = \Pr(Z^*_1 \geq z_1 | N^*_1 = n_1, M^*_1 = m_1) \ldots \Pr(Z^*_k \geq z_k | N^*_k = n_k, M^*_k = m_k),$$

and

$$\Pr(\overrightarrow{Z}_{1:k} \geq \overrightarrow{z}_{1:k} | \overrightarrow{N}_{1:k} = \overrightarrow{n}_{1:k}) = \Pr(Z_1 \geq z_1 | N_1 = n_1) \ldots \Pr(Z_k \geq z_k | N_k = n_k).$$

Therefore to prove (3.6), we need to prove that $\forall j$,

$$\Pr(Z^*_j \geq z_j | N^*_j = n_j, M^*_j = m_j) \geq \Pr(Z_j \geq z_j | N_j = n_j)$$

This is clearly true from Property 3.2.3. □

As an aside, note that Theorem 3.2.2 more generally proves that for any non-decreasing function $f(\overrightarrow{W}_{1:k})$ in $\overrightarrow{W}_{1:k}$, we have that $f(\overrightarrow{Z}^*_{1:k}) \geq^{st} f(\overrightarrow{Z}_{1:k})$.

### 3.6.2 Proof of Theorem 3.2.3

We present a few lemmas before proving the theorem.

**Lemma 3.6.2.** *Given any sequence of non-negative numbers $a_l$, $b_l$ and $c_l$, $l = 1, \ldots, n$. If $\forall\ l, j,\ s.t.\ l > j,\ a_l \geq a_j$ and $\forall h = 1, \ldots, n,\ \sum_{l=h}^{n} b_l \geq \sum_{l=h}^{n} c_l$, then $\sum_{l=1}^{n} a_l b_l \geq \sum_{l=1}^{n} a_l c_l$.*

*Proof.* We know that $\forall h,\ \sum_{l=h}^{n} b_l \geq \sum_{l=h}^{n} c_l$ and $\forall\ l, j,$ st $l > j,\ a_l \geq a_j$. So $\forall h$,

$$(a_h - a_{h-1})(\sum_{l=h}^{n} b_l) \geq (a_h - a_{h-1})(\sum_{l=h}^{n} c_l),$$

where $a_0$ is defined to be equal to 0. Summing over all $h$, we get

$$\sum_{h=1}^{n}\{(a_h - a_{h-1})(\sum_{l=h}^{n} b_l)\} \geq \sum_{h=1}^{n}\{(a_h - a_{h-1})(\sum_{l=h}^{n} c_l)\}.$$

This simplifies to $\sum_{l=1}^{n} a_l b_l \geq \sum_{l=1}^{n} a_l c_l$.

□

**Lemma 3.6.3.** *Consider the static division scheduling scheme with all real-time users being allocated an equal number of $k$ tokens and maximum rate selection criterion. Then under Assumption 3.2.1, the data received by a typical real-time user when it gets served for the last time, i.e., the $k^{th}$ time has the same distribution as $Y$, i.e., $Z_k \sim Y$.*

*Proof.* Due to Property 3.2.1 and 3.2.2, the probability that a user is the first one to leave the system, i.e., be selected for service $k$ times is $1/n$. If it is the first user to leave the system, then $Z_k \sim X^{(n)}$. Similarly for any value of $j$, the probability that a user gets selected the $k^{th}$ time when there are a total of $j$ active real-time users in the system is $1/n$. Then for that user, $Z_k \sim X^{(j)}$. Hence, $Z_k \sim Y$. □

Define a set,

$$S_{\bar{n}_l} = \{\overrightarrow{n}_{l+1:k} | \exists n_j \text{ in } \overrightarrow{n}_{l+1:k} \text{ s.t. } n_j \geq \bar{n}_l \text{ and } \Pr(\overrightarrow{Z}_{l+1:k} \geq \overrightarrow{z}_{l+1:k} | \overrightarrow{N}_{l+1:k} = \overrightarrow{n}_{l+1:k}) \geq \Pr(\overrightarrow{Z}_{l+1:k} \geq \overrightarrow{z}_{l+1:k} | \overrightarrow{N}_{l+1:k} = (\bar{n}_l, \ldots, \bar{n}_l))\}.$$

**Lemma 3.6.4.** *For any* $\overrightarrow{n}_{l+1:k} \notin S_{\bar{n}_l}$,

$$\Pr(\overrightarrow{Z}_{l+1:k} \geq \overrightarrow{z}_{l+1:k} | \overrightarrow{N}_{l+1:k} = \overrightarrow{n}_{l+1:k}) \leq \Pr(\overrightarrow{Z}_{l+1:k} \geq \overrightarrow{z}_{l+1:k} | \overrightarrow{N}_{l+1:k} = (\bar{n}_l, \ldots, \bar{n}_l)).$$

*Proof.* We give the proof by contradiction. Assume that $\exists \overrightarrow{n}_{l+1:k} \notin S_{\bar{n}_l}$ s.t.

$$\Pr(\overrightarrow{Z}_{l+1:k} \geq \overrightarrow{z}_{l+1:k} | \overrightarrow{N}_{l+1:k} = \overrightarrow{n}_{l+1:k}) > \Pr(\overrightarrow{Z}_{l+1:k} \geq \overrightarrow{z}_{l+1:k} | \overrightarrow{N}_{l+1:k} = (\bar{n}_l, \ldots, \bar{n}_l)).$$

Now given the number of users present in the system, the data transferred in a slot is independent of other slots. So,

$$\Pr(\overrightarrow{Z}_{l+1:k} \geq \overrightarrow{z}_{l+1:k} | \overrightarrow{N}_{l+1:k} = \overrightarrow{n}_{l+1:k}) = \Pr(Z_{l+1} \geq z_{l+1} | N_{l+1} = n_{l+1}) \ldots \Pr(Z_k \geq z_k | N_k = n_k)$$

and

$$\Pr(\overrightarrow{Z}_{l+1:k} \geq \overrightarrow{z}_{l+1:k} | \overrightarrow{N}_{l+1:k} = (\bar{n}_l, \ldots, \bar{n}_l)) = \Pr(Z_{l+1} \geq z_{l+1} | N_{l+1} = \bar{n}_l) \ldots \Pr(Z_k \geq z_k | N_k = \bar{n}_l).$$

Since $\overrightarrow{n}_{l+1:k} \notin S_{\bar{n}_l}$, then $\forall j$, $n_j < \bar{n}_l$. So $\forall j$, by Property 3.2.3

$$\Pr(Z_j \geq z_j | N_j = \bar{n}_l) \geq \Pr(Z_j \geq z_j | N_j = n_j).$$

This contradicts our assumption. □

We prove Theorem 3.2.3 now.

*Proof.* The goal is to show $\sum_{j=1}^{k} Z_j \geq^{st} \sum_{j=1}^{k} Y_j$, i.e., $\forall\ z$,

$$\Pr(\sum_{j=1}^{k} Z_j \geq z) \geq \Pr(\sum_{j=1}^{k} Y_j \geq z).$$

To prove this, we will show that for any vector $\overrightarrow{z}_{1:k} = (z_1, \ldots, z_k)$,

$$\Pr(\overrightarrow{Z}_{1:k} \geq \overrightarrow{z}_{1:k}) \geq \Pr(\overrightarrow{Y}_{1:k} \geq \overrightarrow{z}_{1:k}).$$

Since the $Y_j$'s are independent, this simplifies the above inequality to

$$\Pr(\overrightarrow{Z}_{1:k} \geq \overrightarrow{z}_{1:k}) \geq \Pr(Y_1 \geq z_1) \ldots \Pr(Y_k \geq z_k). \tag{3.8}$$

Using conditioning, we can rewrite the left side of (3.8) as

$$\Pr(Z_1 \geq z_1 | \overrightarrow{Z}_{2:k} \geq \overrightarrow{z}_{2:k}) \ldots \Pr(Z_j \geq z_j | \overrightarrow{Z}_{j+1:k} \geq \overrightarrow{z}_{j+1:k}) \ldots \Pr(Z_k \geq z_k).$$

Then (3.8) can be proved if we show that $\forall j$,

$$\Pr(Z_j \geq z_j | \overrightarrow{Z}_{j+1:k} \geq \overrightarrow{z}_{j+1:k}) \geq \Pr(Y_j \geq z_j).$$

Conditioning on the number of users present in the system during the $j^{th}$ time when the user is served,

$$\begin{aligned} &\textstyle\sum_{n_j=1}^{n} \{\Pr(Z_j \geq z_j | \overrightarrow{Z}_{j+1:k} \geq \overrightarrow{z}_{j+1:k}, N_j = n_j) \Pr(N_j = n_j | \overrightarrow{Z}_{j+1:k} \geq \overrightarrow{z}_{j+1:k})\} \\ &\qquad \geq \textstyle\sum_{n_j=1}^{n} \{\Pr(Y_j \geq z_j | \tilde{N}_j = n_j) \Pr(\tilde{N}_j = n_j)\}, \end{aligned} \tag{3.9}$$

where all $\tilde{N}_j$ are i.i.d. and uniformly distributed on $\{1, \ldots, n\}$ (from (3.1)). Given the number of users in a slot, the data obtained is independent of data received in other slots, so

$$\Pr(Z_j \geq z_j | \overrightarrow{Z}_{j+1:k} \geq \overrightarrow{z}_{j+1:k}, N_j = n_j) = \Pr(Z_j \geq z_j | N_j = n_j).$$

Note that when $N_j = n_j$, then a user will have to compete among $n_j$ users to get service in the slot, so

$$\Pr(Z_j \geq z_j | N_j = n_j) = \Pr(X^{(n_j)} \geq z_j).$$

Also from equation (3.1), we have

$$\Pr(Y_j \geq z_j | \tilde{N}_j = n_j) = \Pr(X^{(n_j)} \geq z_j).$$

We can simplify (3.9) to,

$$\begin{array}{c} \sum_{n_j=1}^{n} \Pr(X^{(n_j)} \geq z_j) \Pr(N_j = n_j | \overrightarrow{Z}_{j+1:k} \geq \overrightarrow{z}_{j+1:k}) \geq \\ \sum_{n_j=1}^{n} \Pr(X^{(n_j)} \geq z_j) \Pr(\tilde{N}_j = n_j). \end{array} \tag{3.10}$$

From Lemma 3.6.2, (3.10) can be proved if $\forall l$,

$$\Pr(X^{(l+1)} \geq z_j) \geq Pr(X^{(l)} \geq z_j) \tag{3.11}$$

and $\forall \bar{n}_j$,

$$\Pr(N_j \geq \bar{n}_j | \overrightarrow{Z}_{j+1:k} \geq \overrightarrow{z}_{j+1:k}) \geq \Pr(\tilde{N}_j \geq \bar{n}_j). \tag{3.12}$$

From Property 3.2.3, it is clear that (3.11) is true. To prove (3.12), first consider the right hand side of the equation. Referring to Lemma 3.6.3, we get

$$\Pr(\tilde{N}_j \geq \bar{n}_j) = \Pr(N_k \geq \bar{n}_j) = \textstyle\sum_{\overrightarrow{n}_{j+1:k} | n_k \geq \bar{n_j}} \Pr(\overrightarrow{N}_{j+1:k} = \overrightarrow{n}_{j+1:k}). \tag{3.13}$$

We know that $\forall j$, $N_j \geq N_{j+1}$ almost surely. Thus $\{\overrightarrow{n}_{j+1:k} | n_k \geq \bar{n_j}\} \subseteq S_{\bar{n}_j}$, then from (3.13) we get

$$\sum_{\overrightarrow{n}_{j+1:k} \in S_{\bar{n}_j}} \Pr(\overrightarrow{N}_{j+1:k} = \overrightarrow{n}_{j+1:k}) \geq \Pr(\tilde{N}_j \geq \bar{n}_j). \tag{3.14}$$

Now consider the left hand side of (3.12), conditioning on $\overrightarrow{N}_{j+1:k}$, we get

$$\begin{array}{c} \sum_{\overrightarrow{n}_{j+1:k}} \{\Pr(N_j \geq \bar{n}_j | \overrightarrow{Z}_{j+1:k} \geq \overrightarrow{z}_{j+1:k}, \overrightarrow{N}_{j+1:k} = \overrightarrow{n}_{j+1:k}) \\ \Pr(\overrightarrow{N}_{j+1:k} = \overrightarrow{n}_{j+1:k} | \overrightarrow{Z}_{j+1:k} \geq \overrightarrow{z}_{j+1:k})\} \geq \\ \sum_{\overrightarrow{n}_{j+1:k} \in S_{\bar{n}_j}} \{\Pr(N_j \geq \bar{n}_j | \overrightarrow{Z}_{j+1:k} \geq \overrightarrow{z}_{j+1:k}, \overrightarrow{N}_{j+1:k} = \overrightarrow{n}_{j+1:k}) \\ \Pr(\overrightarrow{N}_{j+1:k} = \overrightarrow{n}_{j+1:k} | \overrightarrow{Z}_{j+1:k} \geq \overrightarrow{z}_{j+1:k})\}. \end{array} \tag{3.15}$$

Note that for $\overrightarrow{n}_{j+1:k} \in S_{\bar{n}_j}$,

$$\Pr(N_j \geq \bar{n}_j | \overrightarrow{Z}_{j+1:k} \geq \overrightarrow{z}_{j+1:k}, \overrightarrow{N}_{j+1:k} = \overrightarrow{n}_{j+1:k}) = 1.$$

So combining (3.14) and (3.15), if we can show that

$$\Pr(\overrightarrow{N}_{j+1:k} \in S_{\bar{n}_j} | \overrightarrow{Z}_{j+1:k} \geq \overrightarrow{z}_{j+1:k}) \geq \Pr(\overrightarrow{N}_{j+1:k} \in S_{\bar{n}_j}),$$

then we would have proven (3.12). Using Bayes' formula we can rewrite the above inequality as,

$$\Pr(\overrightarrow{Z}_{j+1:k} \geq \overrightarrow{z}_{j+1:k} | \overrightarrow{N}_{j+1:k} \in S_{\bar{n}_j}) \geq \Pr(\overrightarrow{Z}_{j+1:k} \geq \overrightarrow{z}_{j+1:k}).$$

Conditioning again on $\overrightarrow{N}_{j+1:k}$, we get

$$\begin{gathered} \textstyle\sum_{\overrightarrow{n}_{j+1:k} \in S_{\bar{n}_j}} \{\Pr(\overrightarrow{Z}_{j+1:k} \geq \overrightarrow{z}_{j+1:k} | \overrightarrow{N}_{j+1:k} = \overrightarrow{n}_{j+1:k}) \\ \Pr(\overrightarrow{N}_{j+1:k} = \overrightarrow{n}_{j+1:k} | \overrightarrow{N}_{j+1:k} \in S_{\bar{n}_j})\} \geq \\ \textstyle\sum_{\overrightarrow{n}_{j+1:k}} \{\Pr(\overrightarrow{Z}_{j+1:k} \geq \overrightarrow{z}_{j+1:k} | \overrightarrow{N}_{j+1:k} = \overrightarrow{n}_{j+1:k}) \\ \Pr(\overrightarrow{N}_{j+1:k} = \overrightarrow{n}_{j+1:k})\}. \end{gathered} \tag{3.16}$$

This is true as a consequence of Lemma 3.6.4. □

Table 3.1: Notation Summary

| | |
|---|---|
| $n$ | Number of real-time users |
| $n_b$ | Number of best effort users |
| $A_r(t)$ | Set of active real-time users during slot $t$ |
| $A_b(t)$ | Set of active best effort users during slot $t$ |
| $A(t)$ | Set of all active users during slot $t$, $A(t) = A_r(t) \cup A_b(t)$ |
| $X^i$ | Marginal distribution of channel capacity seen by User $i$ on a typical slot |
| $x^i(t)$ | Realization of $X^i$ during slot $t$ |
| $F_i(\cdot)$ | Distribution function of $X^i$ |
| $F_i^{-1}(\cdot)$ | Inverse function of $F_i(\cdot)$ |
| $X$ | Under Assumption 3.2.1, $X \sim X^i$ |
| $X^{(l)}$ | $X^{(l)} = \max\{X_1, \ldots, X_l\}$ where $X_j$'s are i.i.d. and $X_j \sim X$ |
| $X^{i,(l)}$ | $X^{i,(l)} = \max\{X_1^i, \ldots, X_l^i\}$ where $X_j^i$'s are i.i.d. and $X_j^i \sim X^i$ |
| $r$ | Minimum rate guarantee given to a real-time user |
| $\tau$ | Time period over which rate guarantee is given |
| $\delta$ | Outage probability of the rate or QoS guarantee |
| $\tilde{N}$ | Uniformly distributed on $\{1, \ldots, n\}$ |
| $Y$ | $Y \sim X^{(\tilde{N})}$, i.e., $Y = X^{(l)}$ $w.p.$ $\frac{1}{n}$, $l = 1, \ldots, n$ |
| $Y^i$ | $Y^i \sim X^{i,(\tilde{N})}$, i.e., $Y = X^{i,(l)}$ $w.p.$ $\frac{1}{n}$, $l = 1, \ldots, n$ |
| $N_j^*$ | Number of contending real-time users when a typical real-time user gets selected for service $j^{th}$ time under the the proposed scheduling scheme and Assumption 3.2.1 |
| $Z_j^*$ | Rate seen by a typical real-time user the $j^{th}$ time he is selected for service under the proposed scheduling scheme and Assumption 3.2.1 |
| $Z_j^{i*}$ | Same quantity as $Z_j^*$, but for User $i$ under Assumption 3.3.1 |
| $N_j$ | Same quantity as $N_j^*$, but for static division scheme |
| $Z_j$ | Same quantity as $Z_j^*$, but for static division scheme |
| $Z_j^i$ | Same quantity as $Z_j^{i*}$, but for static division scheme |
| $k$ | Number of tokens allocated to one/group of real-time user(s) |
| $k^j$ | Number of tokens required by $j^{th}$ real-time user |
| $S$ | A subset of tokens $S \subset \{1, \ldots, k\}$ |
| $\phi(\cdot)$ | Set valued function for the selection criterion |

# Chapter 4

# Reducing Feedback Overhead for Opportunistic Scheduling

## 4.1 Introduction

***Motivation.*** Opportunistic scheduling can lead to significant gains in the performance of wireless networks. However, these gains are achieved at the expense of increased feedback overhead. Whenever a base station makes an opportunistic decision on the user(s) to serve, it needs to know the 'current' channel capacity (or some function of it) for all of the users. Therefore before *each* decision, *all* users need to transmit their current channel conditions to the base station. This can be a high overhead in terms of the bandwidth and the energy expended (especially at the mobile) for feedback, as compared to the gains in throughput that one might hope to glean from opportunistic scheduling.

For example, consider a system where *all* users are experiencing independent and identically distributed Rayleigh channel fading signal to noise ratio (SNR) and all feedback their channel state prior to each data transmission slot. In this case, the gain in the average signal to noise ratio (SNR) of the user selected for service grows roughly logarithmically with the number of users, whereas the amount of feedback overhead increases linearly. Suppose the system has 100 users, then one can achieve 90% of the gain in average SNR by soliciting feedback from a random subset of only 60 users and choosing to serve the user with the highest current SNR. This underscores the need and the potential to reduce the amount of feedback required to realize the major gains of opportunistic scheduling.

One can reduce the resources used for soliciting and transmitting feedback in the following two simple ways.

*Contention.* In the contention based approach users compete for a pool of resources allocated, e.g., CDMA code, a time slot, etc., to feed back their current channel capacity state. For example, a user may opportunistically send its feedback if its current channel capacity is above a certain threshold. The thresholds are designed so that feedback is successful, i.e., only one or a limited number of users contend at the same time. This allows one to achieve a large part of the opportunistic gains possible over long time periods. However, it is possible that feedback gets wasted because too many users contend to transmit their states. Therefore it is possible that occasionally no opportunism is exploited.

*Polling.* Alternatively the base station may solicit feedback from a subset of users, i.e., allocates feedback resources for the subset of users, and choose to opportunistically among them. Since we are exploiting opportunism over only a subset of users the long term gains of this approach are generally reduced as compared to the contention based approach. However, because there is no possibility of wasting feedback resources, some degree of opportunism is almost always exploited.

***Related Work.*** Let us discuss some of the previous work done in this area. A simple threshold based scheme was proposed in [11] to reduce the feedback overhead. In their setup each user had a dedicated resource for sending its feedback. The idea was to only allow a user whose current channel capacity exceeds a threshold to feedback his current state. Their results showed that such a scheme led to a significant reduction in the amount of feedback data while resulting in only a small penalty in the overall throughput. However, even though the energy spent in transmitting and receiving the feedback is reduced, the need for other resource requirements (like bandwidth, etc.) for sending feedback is not reduced.

Some contention based schemes have also been proposed in the literature. One of the frequently cited ideas is 'opportunistic splitting' proposed in [26]. The scheme was proposed in an uplink context, but it is also applicable to downlink scheduling (which is the focus in this chapter). The idea is to divide time into equal sized time units, each unit consists of mini slots which are pooled resources used to learn the current channel capacity of users via feedback, while the rest of the unit is used for data transmission to the selected user.

In opportunistic splitting, initially a pair of thresholds depending on the number of users is set. At the start of the first mini slot, every user whose current channel capacity is between the pair of thresholds contends, i.e., transmits to the base station. The base station then broadcasts to all the users whether on the mini slot no user contended, exactly one user contended, or a collision occurred, i.e., more than one user contended and the base station was unable to decode any information. Depending on the broadcast message received, each user modifies its threshold according to a binary search like algorithm and users' whose channel capacity is between the new thresholds contend in the next mini slot. This process continues until exactly one user contends, therefore the number of mini slots before a transmission may vary. The last user to contend is guaranteed to be the user with the highest current channel capacity. The authors show that on average 2.5 mini slots will be required for the algorithm to find the user with the highest current channel capacity. This is a significant reduction in feedback as compared to the naive scheme requiring a linear number of slots required for soliciting feedback from all users.

However, opportunistic splitting requires two way communication and coordination between the base station and users every mini slot. This may be hard to implement since the time scales involved are quite small. (In practical systems, a data transmission slot is on the order of milliseconds, thus mini slots should be much smaller than milliseconds.) To overcome this

coordination problem, a random access based feedback protocol was proposed in [35], where only one way communication is required. In their scheme each data transmission is preceded by a *fixed* number of mini slots (the smaller the number of mini slots, the lesser the time used in feedback). In each mini slot, users whose current channel capacity exceeds a threshold contend with some probability. If on a given mini slot exactly one user contends, then that user's identity is stored at the base station. Subsequently the base station randomly serves one of the identified users. However if no user is identified, one is selected at random for service. The threshold and probability of contention can be optimized to maximize the overall sum capacity if the channel distributions are known. However, simulation results presented in the paper show that a truncated and thus comparable version of opportunistic splitting usually performed better than the proposed scheme. (Some researchers have also studied reducing feedback overhead in OFDM systems [30][34], which is not the focus here.)

An underlying assumption in the above research was that users in the system see i.i.d. channel capacity distributions. (Note that an extension of opportunistic splitting to the case where users can experience one of two possible channel capacity distributions is presented in [27], however this still does not seem a reasonable model.) In practice users' channel capacity variations are heterogenous, e.g., users close to a base station see significantly different channel capacity than those further off. Extending these schemes to the non i.i.d. case is in general very complex, because the thresholds then will not only be dependent on the user's own channel capacity distribution and the number of users, but also other users' channel capacity distributions. Ideally one would like to have an easy way to set the thresholds for the heterogeneous case.

All of the above mentioned work focussed on reducing feedback in the context of opportunistic scheduling of best effort traffic. Whereas base stations are likely to support a mixture of both best effort and real-time traffic. Real-

time traffic has requirements over short time scales, therefore for real-time traffic to benefit from opportunism one needs to exploit opportunism over short time scales. Additionally, as the deadline for meeting users' quality of service (QoS) requirement approaches, the base station needs to serve only those users whose deadline is nearing. Therefore the base station can exploit opportunism among only the subset of users that are roughly in equal danger of not meeting their QoS requirements. In other words, in a real-time context the requirements are more stringent in the sense that opportunism needs to be exploited over short time scales, so there are lesser opportunities to do so. Therefore the need for efficient utilization of feedback resources is higher as compared to the best effort case.

***Contributions.*** The following are the contributions of this chapter.

We first propose a contention based scheme, which we shall call *static splitting*, also geared at reducing feedback overheads in a best effort traffic only scenario. Like [35], our setup consists of a fixed number of mini slots, and does not require two way communication. We will combine static splitting with maximum quantile scheduling to handle heterogeneity in users' channel capacity variations. This allows one to compute a common threshold determining when users are to transmit feedback, which is *independent of their possibly heterogenous channel distributions*. This is unlike other proposed schemes as it allows off-line calculation of 'optimal' thresholds. Static splitting is designed to find a user that is currently experiencing a high quantile, instead of finding the users with the highest quantile (which may require a lot more resources). The advantage of such an approach is supported by our simulation results which indicate that static splitting can perform much better (for example 40% improvement) than a truncated form of opportunistic splitting.

We then consider a scenario where traffic is a mixture of best effort and real-time traffic, for which QoS guarantees have to be met over short time scales. We argue that in such a scenario a combination of contention

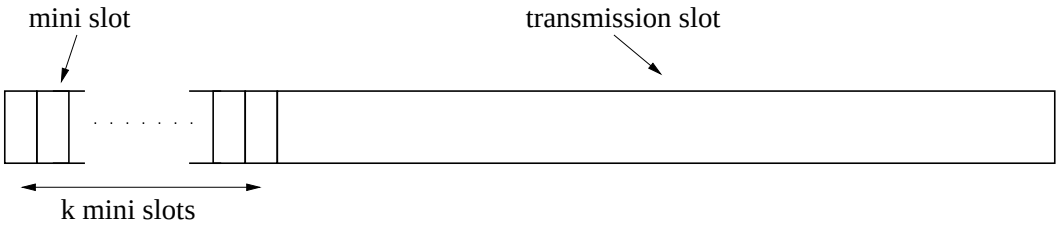



Figure 4.1: Structure of a time unit.

and polling based feedback strategies is needed. Based on this insight, we combine static splitting and dynamic polling with a variation of a token based scheduling scheme proposed in Chapter 3 to provide QoS. We call this the *joint polling and opportunistic scheduling* (JPOS) scheme. Under fast fading, we show a lower bound on the service seen by a real-time user under the JPOS scheme. Furthermore, based on this scheme we propose a heuristic that simulations indicate not only meets users' QoS guarantees, but achieves up to 89% of system capacity in terms of the long term throughput that is realized.

*Chapter Organization.* The chapter is organized as follows. In Section 4.2 we describe the proposed static splitting scheme in the best effort traffic only scenario. In Section 4.3 we consider the case where traffic is a mixture of both best effort and real-time flows and describe the proposed JPOS scheme to reduce feedback in such mixed scenario. Simulation results are presented in Section 4.4, and Section 4.5 concludes this chapter.

## 4.2 Opportunistic Feedback Based on Static Splitting

### 4.2.1 System Model and Notation

We begin by introducing our system model and some notation. Like the previous two chapters we consider only downlink scheduling for simplicity. The system model is slightly different from the previous chapters. Here time is divided into equal size 'time units'. Each time unit consists of $k$ equal size mini slots followed by a transmission slot during which at most one user can be served (see Figure 4.1). The $k$ mini slots are used for collecting feedback. We

will define the exact nature of the feedback later. Following are the assumption on user's channel capacity characteristics over time units.

**Assumption 4.2.1.** *We assume the channel capacity (rate) for each user is a stationary ergodic process and these processes are independent, but not necessarily identically distributed across users. Further we assume that the marginal distribution for each user is continuous and either known a priori (or estimated) at the user.*

We have justified most of the assumptions in the previous two chapters. We note that the assumption that a user has a priori knowledge of the marginal distributions for the channel capacity is implicitly used in all of the previous work discussed above [35][11][27][26]. We require the users' rates to be continuous only to keep our discussion simple. (In fact we perform our simulations for discrete rate distributions.) The details on handling the discrete case were discussed in Chapter 2.

The notation is also similar to that in previous chapters. We will let $x^i(t)$ denote the realization of the downlink channel capacity/rate of User $i$ at time unit $t$, and let $X^i$ be a random variable whose distribution is that of the channel capacity of User $i$ on a *typical* time unit. Recall that we assume $X^i$ to be continuous random variables that are independent but need not necessarily be identically distributed across users. We denote the distribution function of $X^i$ by $F_{X^i}(\cdot)$. For simplicity, we will assume that $F_{X^i}(\cdot)$ is a strictly increasing function, so that its inverse denoted by $F^{-1}_{X^i}(\cdot)$ is defined. Finally note that by Assumption 4.2.1 $F_{X^i}(\cdot)$ is known at the user.

For analysis purposes, in this section we will only consider a 'fixed saturated' regime where there is a fixed number of users in the system and each user in the system is infinitely backlogged. For now we allow only best effort flows. The number of users in the system are denoted by $n$.

### 4.2.2 Proposed Static Splitting Feedback Scheme for Best Effort Traffic

In Chapter 2 we discussed that maximum quantile scheduling has very desirable properties, it maximizes the amount of opportunism, i.e., quantile of the user being served, is intrinsically temporally fair, and asymptotically in the number of users, maximizes the sum throughput. As compared to other opportunistic scheduling schemes proposed in literature, e.g., maximum throughput [16], proportionally fair [2], etc., maximum quantile has some distinct advantages in handling cases where users have heterogeneous, unknown and possibly slowly varying channel capacity distributions. In particular when one must resort to estimating parameters, such schemes tend to lose in terms of throughput and fairness. Therefore, in this chapter, we will focus on channel feedback schemes which are compatible with opportunistic scheduling of users currently experiencing channel capacity in the high quantile, i.e., high $F_{X^i}(x^i(t))$.

Recall that in each time unit the data transmission part is preceded by $k$ mini slots. The objective during the mini slots is to identify a user whose current rate is in a high quantile. Note that we do not necessarily have to identify the user with the highest quantile, we will revisit this point later. For simplicity we will assume in this subsection that the number of users $n$ is such that $\frac{n}{k}$ is an integral value. In our proposed scheme, each user is associated with exactly one mini slot, so $\frac{n}{k}$ users are associated with each mini slot. A user can only contend (i.e., send feedback) on the mini slot with which it is associated. In other words, users are split into $k$ 'static' groups, and users within the same group may contend for the same mini slot. Based on $n$ and $k$ (which a user learns from the base station), each user calculates (looks up) a quantile threshold denoted by $q^i$, $i = 1, \ldots, n$ which is used to determine if it will contend by transmitting feedback – we will give details on optimizing $q^i$ later. Specifically, recall that at time unit $t$, the rate User $i$ can support is denoted by $x^i(t)$. Prior to User $i$'s mini slot the user would check

if $x^i(t) > F_{X^i}^{-1}(q^i)$, and if so it would transmit the quantile $F_{X^i}(x^i(t))$ of its current rate to the base station.

If only one user contends during a given mini slot, we assume that the base station is able to determine both the identification of the user and the value of its quantile, and store this information. If more than one user contends for a given mini slot, a collision occurs, and the base station stores this fact. Finally, if no user contends for the mini slot, then no action is taken by the base station. The process is repeated across all mini slots.

Once the contention for mini slots is finished, if the base station was able to identify at least one user, then it serves the user with the highest quantile among the identified users. Else, if the base station fails to identify any such user, then it serves a randomly selected user. This can occur in two ways, if collisions have been recorded for none of the mini slots, then a user is randomly selected from all the users. However, if a collision occurred on at least one of the mini slots, then a user is randomly selected for service among the groups of users associated with the mini slots where collisions occurred. Doing so, increases the chance of choosing a user with a high quantile.

The last challenge for this simple protocol is determining a good choice for the contention thresholds $q^i, i = 1, \ldots, n$. Below we will do a simple analysis assuming a common threshold $q$ across users, i.e., $q^i = q$, $\forall i = 1, \ldots, n$. Let $A^i$ denote the event that User $i$ is selected under the above protocol and $\mathbf{1}_{A^i}$ be the indicator function for $A^i$. Our goal is to serve users with a high quantile, i.e., maximize the expected *sum quantile* $E[\sum_{i=1}^n F_{X^i}(X^i)\mathbf{1}_{A^i}]$ of the scheduled users. Recall that we have assumed $\frac{n}{k}$ is integer valued, i.e., each mini slot has exactly the same number of users, and $F_{X^i}(X^i)$ are i.i.d. across users, thus if all users share a common threshold $q$, it follows by symmetry that each user is equally likely to be selected, i.e., $\Pr(A^i) = \frac{1}{n}$, and so

$$E[\sum_{i=1}^{n} F_{X^i}(X^i)\mathbf{1}_{A^i}] = \sum_{i=1}^{n} E[F_{X^i}(X^i)|A^i]\Pr(A^i) = \frac{1}{n}\sum_{i=1}^{n} E[F_{X^i}(X^i)|A^i].$$

Again by symmetry, $E[F_{X^i}(X^i)|A^i]$ is equal across all users, so it suffices to optimize $q$ to maximize this quantity for any user.

Recall that under static splitting User $i$ may be selected for service either because it successfully contended on its mini slot and has the highest quantile among users that were identified, or no user was successful and it was selected at random. Let $A_b^i$, $b = 1, \ldots, k$ denote the event that successful contention occurs over $b$ mini slots, and User $i$ is selected for service. In other words under event $A_b^i$, not only did User $i$ have the highest quantile among its group and was successfully identified by the base station, it also had the highest quantile among the exactly $b$ users that were identified by the base station, and was therefore selected for service. We shall let $A_0^i$ denote the event that User $i$ is selected at random in the case where the feedback protocol was not able to identify any user. Note that $A_b^i$, $b = 1, \ldots, k$ and $A_0^i$ form a partition of $A^i$ and so we have that

$$E[F_{X^i}(X^i)|A^i] = \sum_{b=1}^{k} E[F_{X^i}(X^i)|A_b^i] \Pr(A_b^i|A^i) + E[F_{X^i}(X^i)|A_0^i] \Pr(A_0^i|A^i). \tag{4.1}$$

Let $p_n$ denote the probability that a user is able to successfully contend in a mini slot in a time unit with $n$ competing users, then $p_n = \frac{n}{k}(1-q)q^{\frac{n}{k}-1}$. Now consider $\sum_{i=1}^{n} \Pr(A_b^i)$, it is the total probability of selecting a user that has the highest quantile among the exactly $b$ identified users. This is equal to the probability that the base station identifies exactly $b$ users, i.e., successful contention on $b$ mini slots, then,

$$\sum_{i=1}^{n} \Pr(A_b^i) = \binom{k}{b} (p_n)^b (1-p_n)^{k-b}.$$

Now by symmetry, one can conclude that for any User $i$

$$\Pr(A_b^i) = \frac{1}{n} \binom{k}{b} (p_n)^b (1-p_n)^{k-b}.$$

Furthermore since $\Pr(A^i) = \frac{1}{n}$,

$$\Pr(A_b^i|A^i) = \binom{k}{b} p_n^b (1-p_n)^{k-b}.$$

One can also easily show that

$$E[F_{X^i}(X^i)|A_b^i] = \frac{b}{b+1}(1-q) + q.$$

Finally, we have that

$$\Pr(A_0^i|A^i) = 1 - \sum_{b=1}^{k} \Pr(A_b^i|A^i),$$

and we can approximate $E[F_{X^i}(X^i)|A_0^i]$ as

$$E[F_{X^i}(X^i)|A_0^i] \approx \frac{1}{2},$$

i.e., the average quantile of the selected user when it is selected completely randomly, i.e., ignoring conditioning on the base station being unable to identify any user.

Now putting these results together we can rewrite (4.1) as

$$E[F_{X^i}(X^i)|A^i] \approx \sum_{b=1}^{k} \binom{k}{b} p_n^b (1-p_n)^{k-b} \left(\frac{b}{b+1}(1-q)+q\right) + \frac{1}{2}(1-p_n)^k. \quad (4.2)$$

A good threshold $q$ should maximize the above approximation. This can be carried out numerically by searching over $q \in [0,1]$. In Table 4.1 we list the optimum threshold $q$ for an increasing number of users for $k = 2, \ldots, 9$. The threshold increases with $n$ for a given $k$, and with $k$ for a given $\frac{n}{k}$, as might be expected.

Note that (4.2) is independent of users' channel capacity distributions and can also be used to find an approximate value of $q$ even when $\frac{n}{k}$ is not an integral value. As mentioned earlier, unlike other schemes, this eliminates the need for online real-time calculations, it is sufficient to do off-line calculation of thresholds and store them in a table.

Some final comments, note that opportunistic splitting was designed to find the user with the highest quantile/rate. In a practical system where the number of mini slots may be limited, if the scheme is unable to find the user

Table 4.1: Optimum values of quantile threshold $q$

| $\frac{n}{k}$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| $k=2$ | 0 | 0.6796 | 0.7591 | 0.8064 | 0.8380 | 0.8606 | 0.8777 |
| $k=3$ | 0 | 0.6931 | 0.7689 | 0.8134 | 0.8432 | 0.8647 | 0.8810 |
| $k=4$ | 0 | 0.7069 | 0.7791 | 0.8211 | 0.8491 | 0.8693 | 0.8847 |
| $k=5$ | 0 | 0.7205 | 0.7895 | 0.8291 | 0.8554 | 0.8744 | 0.8888 |
| $k=6$ | 0 | 0.7337 | 0.7998 | 0.8372 | 0.8620 | 0.8798 | 0.8934 |
| $k=7$ | 0 | 0.7462 | 0.8097 | 0.8452 | 0.8686 | 0.8854 | 0.8981 |
| $k=8$ | 0 | 0.7580 | 0.8191 | 0.8530 | 0.8752 | 0.8910 | 0.9030 |
| $k=9$ | 0 | 0.7690 | 0.8279 | 0.8604 | 0.8815 | 0.8965 | 0.9078 |

with the highest quantile in those many mini slots, a user has to be chosen at random. This is not desirable. Whereas if static splitting is unsuccessful in finding the user with the highest quantile, it is still likely to serve a user with high quantile. The possibility of serving a high quantile user is captured in (4.2), in fact the expression also captures the performance of the scheme even when a user is selected at random. Therefore by maximizing (4.2), one can obtain better performance as compared to *truncated* opportunistic splitting especially for small values of $k$. We will verify this using simulations in Section 4.4.

## 4.3 Reducing Feedback for Scheduling Schemes providing Quality of Service

As discussed in Section 4.1, when attempting to ensure quality of service to flows one has to exploit opportunism over small time scales and exploit opportunism among only those users that are roughly in the same danger of not meeting their QoS requirements. (Note that this requires the availability of feedback from users that are roughly in equal danger of not meeting their QoS requirements.) Polling based feedback mechanisms meet these criteria. In particular one can exploit opportunism over short time scales, and by dynamically deciding from whom to solicit feedback, one can exploit opportunism

only among a desired subset of users.

However as mentioned earlier, a polling based approach is not as efficient at exploiting opportunism as a contention based approach. Therefore it makes sense to use contention based static splitting for scheduling best effort users and polling for scheduling of real-time users. We propose a modified form of the scheduling scheme proposed in Chapter 3 that is compatible with such a feedback strategy. Furthermore we improve on this scheme by proposing a heuristic.

First let us modify our system setup to include real-time traffic flows. In our new setup each user is either associated with a real-time or a best effort stream, but not both. The total number of users is still denoted by $n$, while the number of real-time users will be denoted by $n_r$. For simplicity consider the case where both real-time and best effort users are infinitely backlogged.

The notion of QoS considered in this chapter is the same as in Chapter 3, i.e., ensuring a User $i$ sees a desired rate $r^i$ over a frame of length $\tau$ with an outage probability of $\delta^i$. In other words, we divide time into equal sized frames consisting of $\tau$ time units and our goal is to ensure that for each of these frames

$$\Pr(R^i(\tau) \geq r^i) \geq 1 - \delta^i,$$

where $R^i(\tau)$ is a random variable denoting the cumulative rate seen by User $i$ during a frame of length $\tau$ time units.

Like Chapter 3, we will stochastically lower bound the actual service $R^i(\tau)$ by a quantity $\underline{R^i}(\tau)$, i.e.,

$$R^i(\tau) \geq^{st} \underline{R^i}(\tau).$$

We make an additional assumption here on users' rate in this section. We assume that users' channel capacity is fast fading, i.e., for any User $i$ the realizations of $X^i$'s are independent across slots.

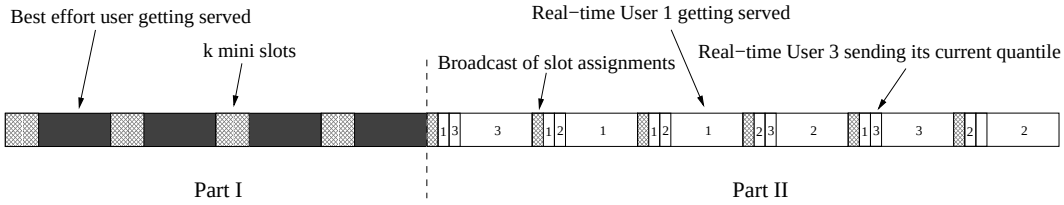



Figure 4.2: Example of the JPOS scheme with frame size of 10 time units and 3 real-time users having 2 tokens each.

### 4.3.1 Joint Polling & Opportunistic Scheduling Scheme

Recall that in the token scheme proposed in Chapter 3 the idea was to opportunistically serve each real-time user exactly $l$ times in every frame of $\tau$ units. This could be thought of as allocating each real-time user $l$ tokens at the start of each frame. Whenever a real-time user was served, its token count goes down by 1, and when it has been served $l$ times it is no longer allowed to contend for service. The JPOS scheme described below also allocates $l$ tokens to each real-time user at the start of each frame, and as a result each real-time user is served exactly $l$ times in each frame.

We describe the scheme with an example. Consider a frame size of $\tau = 10$ time units, where each time unit contains $k = 3$ mini slots. The system has 5 best effort users and 3 real-time users, each real-time user has $l = 2$ tokens. We illustrate a realization of the scheme for the example in Fig. 4.2.

In order to serve each real-time user $l$ times, we divide each frame into two parts. The first part of the frame consists of $\tau - n_r l$ time units and only best effort users are served here. Every best effort user contends for service in every time unit of the first part via the static splitting with maximum quantile scheduling mechanism described in Section 4.2. In our example the first part of the frame consists of 4 time units, where best effort users are served (Fig. 4.2).

During the second part of the frame consisting of $n_r l$ time units, only real-time users are served. Unlike the first part of the frame, here the feedback

mechanism is based on polling. In a time unit the base station decides to solicit the quantile of the current rate from $k-1$ real-time users that currently have *the highest remaining number of tokens*. Ties among real-time users with equal remaining tokens are broken randomly. If there are less than $k-1$ real-time users that currently have a positive token count, then all users are solicited for feedback. The first mini slot of each time unit is used to broadcast which mini slot has been assigned to which real-time user for polling its feedback. The remaining $k-1$ mini slots are used to get feedback on users' quantile information. Note that in this case at most one user will contend in each mini slot, so there is no possibility of a collision. The base station serves the real-time user with the highest quantile among those from which feedback was solicited, and the token count for that real-time user goes down by 1. This process continues until the end of the frame. It should be clear that any real-time user that has been served $l$ times is not allowed to contend any longer, by the end of the frame all real-time users would have been served $l$ times.

Returning to our example, let us describe scheduling in the second part of the frame (Fig. 4.2). The second part starts with the $5^{th}$ time unit. During the $5^{th}$ time unit, all real-time users have $l = 2$ tokens each. Using random tie breaking, the base station chooses to solicit feedback from real-time Users 1 and 3. Since real-time User 3 currently has the higher quantile, it gets served and its token count goes down. In the $6^{th}$ time unit because real-time Users 1 and 2 have a higher remaining token count of 2, feedback is solicited from them, and real-time User 1 (on account of its higher current quantile) gets served. In the $7^{th}$ time unit using random tie breaking among real-time Users 3 and 1, feedback is solicited from real-time Users 2 and 1, and real-time User 1 gets served. Now since real-time User 1 has been served $l = 2$ times, it will no longer be served. In the next time unit real-time Users 2 and 3 get polled, and real-time User 2 is served. In the $9^{th}$ time unit real-time User 3 gets served and is no longer considered for service. Finally in the $10^{th}$ time unit the only remaining user, i.e., real-time User 2 is polled and served. Note that this is

only one of the realization of the scheme the scheme could have proceeded in multiple ways.

Some comments on the proposed scheme. Note that by choosing to poll users that have the highest remaining number of tokens we are not only polling users that have the highest danger of not meeting their QoS requirements, but also are keeping as many real-time users in the system as possible. This allows one to exploit a larger amount of opportunism as compared to the case where some users leave early.

The above scheme ensures QoS by serving each user exactly $l$ times. In doing so, it exploits opportunism available among the best effort users in the first part of the frame, and exploits opportunism among real-time users in the second part of the frame. In other words the scheme is able to exploit intra class opportunism.

The disadvantage of the proposed scheme is that unlike the scheme proposed in Chapter 3 it does not exploit inter class opportunism among best effort and real-time users. We will overcome this by proposing a heuristic based on the above algorithm later, which is similar to the scheme proposed in Chapter 3. Let us next describe how one might determine the value of $l$, the number of tokens per real-time flow.

### 4.3.2 Analysis and Resource Allocation

We now lower bound the service seen by a real-time user under JPOS, this in turn will allow us to conservatively estimate the value of $l$ needed to meet users' QoS requirements.

We have designed our JPOS scheme so that the service seen by a real-time user satisfies the three properties (Equal Resource Allocation, Symmetric Selection & Monotonicity) described in Chapter 3. It is clear that since all real-time users are allocated $l$ tokens each, Property 3.2.1 is satisfied. Property 3.2.2 is trivially satisfied in the first part of the frame. In the second

part of the frame, all users start with an equal number of tokens. Furthermore in each time unit, among real-time users having an equal token count there is random tie breaking in deciding whom to poll, and the quantile of all real-time users are uniformly distributed. From these three symmetrical conditions one can show that Property 3.2.2 holds for the second part of the frame. Let $X^{i,(m)}$ be the maximum of $m$ i.i.d. copies of $X^i$, i.e., $X^{i,(m)} := \max[X_1^i, \ldots, X_m^i]$, where $X_j^i \sim X^i$, $\forall j = 1, \ldots, m$. To show that Property 3.2.3 holds, consider a time unit in the second part of the frame with $m$ competing real-time users. If User $i$ gets selected, then if $m \geq k-1$ it sees a service of $X^{i,(k-1)}$, else it sees a service of $X^{i,(m)}$. (Here $X^{i,(m)}$ is the maximum of $m$ i.i.d. copies of $X^i$.) It is easy to show that Property 3.2.3 is satisfied here.

It should be clear that since all the three properties are satisfied, the bounds developed in the previous chapter can also be developed here. To do so, we introduce some notation. Let $Z_j^{i*}$ be the random variable denoting the rate received by real-time User $i$ conditioned on it getting selected for service the $j^{th}$ time under the above described JPOS scheme. Then the total service seen by real-time User $i$ under the JPOS scheme is given by $\sum_{j=1}^{l} Z_j^{i*}$. Now define a mixture random variable $Y^i$

$$Y^i = \begin{cases} X^{i,(k-1)} \text{ w.p. } 1 - \frac{k-2}{n_r} \\ X^{i,(k-2)} \text{ w.p. } \frac{1}{n_r} \\ \ldots \quad \text{w.p.} \quad \ldots \\ X^{i,(1)} \text{ w.p. } \frac{1}{n_r}. \end{cases} \tag{4.3}$$

One can think of $Y^i$ as User $i$ getting selected for service in the second part of the frame when competing with greater than or equal to $k-1$ real-time users with probability $1 - \frac{k-2}{n_r}$, or getting selected for service when competing with $k-2$ users with probability $\frac{1}{n_r}$ and so on. Note that $Y^i$ only depends on $X^i$ and $n_r$, but does not depend on the channel rate distribution of other users.

We now show that $\sum_{j=1}^{l} Z_j^{i*} \geq^{st} \sum_{j=1}^{l} Y_j^i$, where $Y_j^i$'s are i.i.d. and $\forall j = 1, \ldots, l$, $Y_j^i \sim Y^i$. In other words, the service seen by User $i$ under the

JPOS scheme can be lower bounded by a sum of *i.i.d.* random variables that depends only on the number of real-time users and $X^i$, and yet factors in the opportunism that can be exploited. In fact, we show a stronger bound, i.e., for any set $S \subseteq \{1, \ldots, l\}$, $\sum_{j \in S} Z_j^{i*} \geq^{st} \sum_{j \in S} Y_j^i$. The following theorem formally states our claim. Since the proposed low feedback scheme satisfies all the three properties described above, the proof of the theorem follows from that of Theorem 3.2.3 in the previous chapter.

**Theorem 4.3.1.** *Consider the JPOS scheme where all $n_r$ real-time users are allocated an equal number $l$ of tokens. Then under Assumption 4.2.1 and fast fading on users' channel capacities, for any real-time User $i$*

$$\sum_{j \in S} Z_j^{i*} \geq^{st} \sum_{j \in S} Y_j^i,$$

*for $S \subseteq \{1, \ldots, l\}$. Here $Y_j^i$'s are i.i.d. and $Y_j^i \sim Y^i$, $\forall j = 1, \ldots, l$.*

We can now follow a similar process as described by (3.3) and (3.4) in the previous chapter. If the number of tokens allocated to a user $l'$ is large enough, then the distribution of $\sum_{j=1}^{l'} Y_j^i$ can be roughly approximated, e.g., using the Central Limit Theorem. Then since each real-time user knows its distribution and can learn the value of $n_r$ from the base station, it can calculate its required number of tokens $l^i$ as

$$l^i = \min_{l'} \{l' \mid \Pr(\frac{\rho}{\tau} \sum_{j=1}^{l'} Y_j^i \geq r^i) \geq 1 - \delta^i\}, \tag{4.4}$$

here $\rho$ is the fraction of time unit that is used for data transmission. The value of $l^i$ can be communicated to the base station and the base station can allocate each real-time user $l$ tokens, where

$$l = \max_{i=1,\ldots,n_r} l^i, \tag{4.5}$$

and allocate each real-time user $l$ tokens. Note that we require that, $n_r l \leq \tau$, i.e., the total number of tokens allocated must be less than or equal to the size of frame.

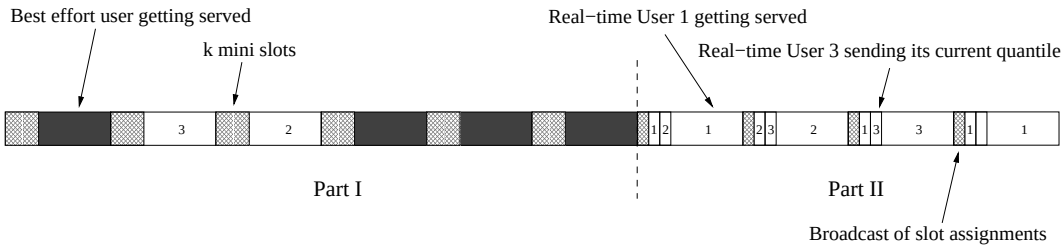



Figure 4.3: Example of the heuristic based JPOS scheme with frame size of 10 time units and 3 real-time users having 2 tokens each.

Note that there is some overhead involved in transmitting the value of $l^i$ from each real-time to the base station. However, this overhead is needed only when the number of real-time users change or a real-time user's channel distribution changes so much that its token requirement changes.

Finally note that it is possible that even if a real-time user requires fewer than $l$ tokens, it is still allocated $l$ tokens. This may seem too conservative, however one can group users together (as descried in the previous chapter) to reduce the total number of tokens required. This completes the description of the JPOS scheme.

### 4.3.3 Heuristic based on Joint Polling & Opportunistic Scheduling Scheme

As discussed earlier the proposed JPOS scheme only exploits the intra class opportunism among the best effort and real-time users, but not the inter class opportunism among all the users. In this subsection we present a modified form of the JPOS scheme which exploits both inter and intra class opportunism and thus achieves a higher overall throughput. This modification is more similar to the token scheme proposed in the previous chapter.

Recall that our goal of polling real-time users was to ensure that users with roughly equal danger of not meeting their QoS requirements are able to send their feedback and opportunism was exploited among them. However, in our proposed scheme the danger of not meeting the QoS requirement only

occurs if the total remaining number of token is equal to (or less) than remaining time units in the frame. Otherwise, there is leeway to exploit opportunism across all users, and this can be exploited by allowing real-time users to compete with best effort users during the first part of the frame. We use this idea in the modification described below.

Like the JPOS scheme, in the proposed modification each real-time user is allocated $l$ tokens at the start of each frame. Whenever a real-time user is served, its token count goes down by 1 and when it has been served $l$ times, it is no longer considered for service. However unlike the original scheme, we allow both the best effort and real-time users to compete using maximum quantile based static splitting during the first part of the frame. Furthermore the size of the first part is dynamic and it lasts until the *total number of remaining tokens in the system is equal to the number of remaining time units in the frame.* Then the second part of the frame starts, here like the JPOS scheme only real-time users are served using polling. The method of deciding the real-time users that are to be polled for their current quantile is the same as the original scheme, i.e., polling real-time users with the $k-1$ highest remaining tokens counts and using random tie breaking for users with equal remaining token count. As before, the real-time user with the highest quantile among those polled is served.

To illustrate the scheme more clearly, we use the example described in Subsection 4.3.1 to describe the original scheme. A realization of the scheme is illustrated in Figure 4.3. Since real-time users are allowed to contend in the first part of the frame, real-time Users 3 and 2 get served in the $2^{nd}$ and $3^{rd}$ time units respectively. As a result, the second part of the frame starts at the $7^{th}$ time unit. Therefore, inter class opportunism is exploited for a longer period of time. At the $7^{th}$ time unit real-time User 1 is polled because it has the highest remaining token count, while real-time User 2 is chosen for polling due to tie breaking. Since real-time User 1 has a higher quantile than real-time User 2, it gets served. The rest of the scheme proceeds as shown in the figure.

Even though the proposed modification scheme does exploit both the inter and intra class opportunism it is difficult to bound the QoS seen by a real-time user. This is because it is not clear whether the Monotonicity property holds under the scheme. (One can show that the Symmetric Selection and Equal Resource Allocation properties do hold.) However, we conjecture that calculating the value of $l$ according to (4.4) and (4.5) will allow us to meet the required QoS guarantees. This conjecture is supported by the simulation results presented in the next section. These also confirm the superior overall throughput performance of the proposed modification compared to the JPOS scheme.

## 4.4 Simulation Results

In this section we present simulation results for the schemes proposed in this chapter. Let use describe the general simulation setup we used. In order to compare the performance of our scheme to other schemes discussed in literature, we assumed that all users undergo i.i.d. Rayleigh fast fading with a mean SNR of 2. We used the SNR to rate mapping as done in CDMA-HDR [2].

### 4.4.1 Static Splitting Performance

We first simulated static splitting for the best effort traffic only scenario. The number of users associated with a mini slot were varied from 1 to 7, while $k = 2, 4, 6, 8$ mini slots were used. The threshold $q$ was set as discussed in Section 4.2.

For comparison, we also simulated a truncated form of opportunistic splitting using the above described setup. However, note that a mini slot in opportunistic splitting consists of two transmissions, whereas a mini slot in our scheme consists of only one transmission. Therefore to be fair, we compare our scheme with a truncated form of opportunistic splitting where at most $\frac{k}{2}$

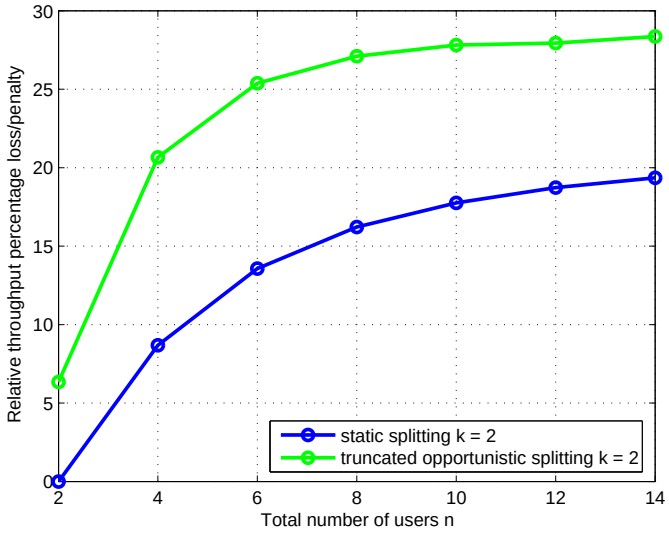



Figure 4.4: The relative percentage throughput loss due to static splitting and a truncated form of opportunistic splitting for $k = 2$ mini slots and an increasing number of users.

mini slots are used. At the end of $\frac{k}{2}$ mini slots if the algorithm is unable to find the user with the highest quantile, then it selects a user at random.

We compared the throughput achieved by all the schemes to that achieved by a virtual scheme that is able to select the user with the highest rate every time unit. This is the best that the schemes can hope to achieve. We plot our results as the relative percentage loss in throughput compared to that achieved by the virtual scheme in Figures 4.4, 4.5, 4.6 and 4.7. Note that as expected the relative penalty for both the schemes goes down with increasing values of $k$. Furthermore observe that the relative penalty for both schemes increases with the number of users.

The results illustrate the advantage of using static splitting, our scheme does better than opportunistic splitting for $k = 2, 4, 6$. The difference in performance can be significant, for example at $k = 4$ and $n = 12$ in Figure 4.5, the relative loss for static splitting is 9.63%, while the loss for opportunistic splitting is 15.93%. Therefore static splitting reduces the loss by about 40%.

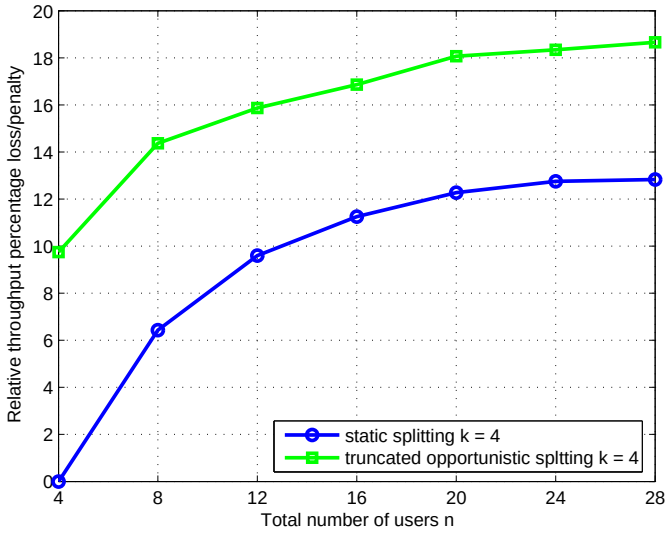



Figure 4.5: The relative throughput percentage loss due to static splitting and a truncated form of opportunistic splitting for $k = 4$ mini slots and an increasing number of users.

Note that for $k = 6$ truncated opportunistic splitting has $\frac{k}{2} = 3$ mini slots to find the 'best' user, this is greater than the average of 2.5 slots needed for the scheme. Even there static splitting does better.

At $k = 8$ in Figure 4.7, opportunistic splitting starts to do better than static splitting. This is expected because as $k$ increases, opportunistic splitting is increasingly able to find the user with the highest rate. However, note that the engineering complexity needed for opportunistic splitting may not justify the gain it shows over static splitting.

### 4.4.2 Performance of Joint Polling & Opportunistic Scheduling Scheme

As a second experiment we simulated the proposed JPOS scheme and its heuristic modification. In our setup there were 12 best effort users and 12 real-time users. Each time unit consisted of $k = 6$ mini slots and was 5 msec long. The data transmission part of each time unit was 4.5 msec long (i.e., $\rho = 0.9$) with the rest being used to gather feedback. Each real-time user was

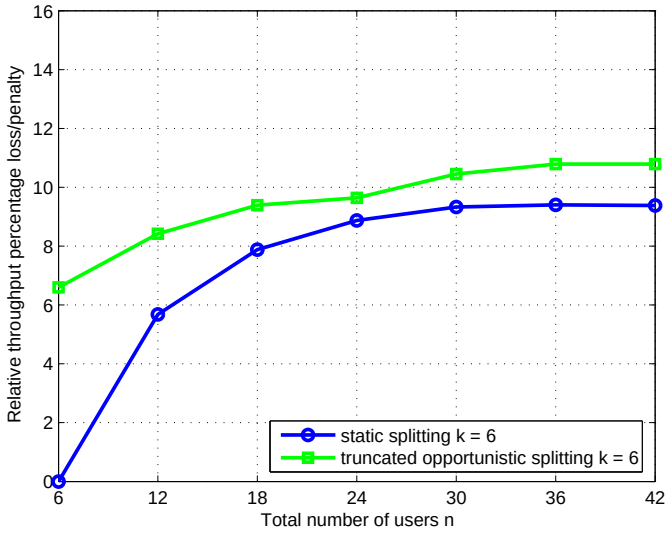



Figure 4.6: The relative throughput percentage loss due to static splitting and a truncated form of opportunistic splitting for $k = 6$ mini slots and an increasing number of users.

given a guarantee of 40 kbps with an outage probability of 1% over frame size varying from 100 time units to 600 time units in steps of 100 time units (i.e., 500 msec to 3 sec in steps of 500 msec).

We kept track of the throughput achieved by the schemes and whether the real-time users were able to meet their guarantee. As expected the JPOS scheme was able to provide the required guarantee to all the real-time users. Additionally, the heuristic modification was also able to provide the required guarantee to all the real-time users in all the cases. This supports our conjecture that the heuristic modification will be able to meet real-time users' QoS requirements. We again compared the throughput achieved by the schemes to a virtual scheme that always serves the user with the highest current rate. The results as a percentage of throughput achieved by the virtual scheme are plotted in Fig 4.8. Note that the throughput for both schemes increases as the QoS guarantee is given over longer time frames. We also observe that the heuristic modification has a higher overall throughput, clearly illustrating the advantage of exploiting inter class opportunism. Furthermore, both schemes

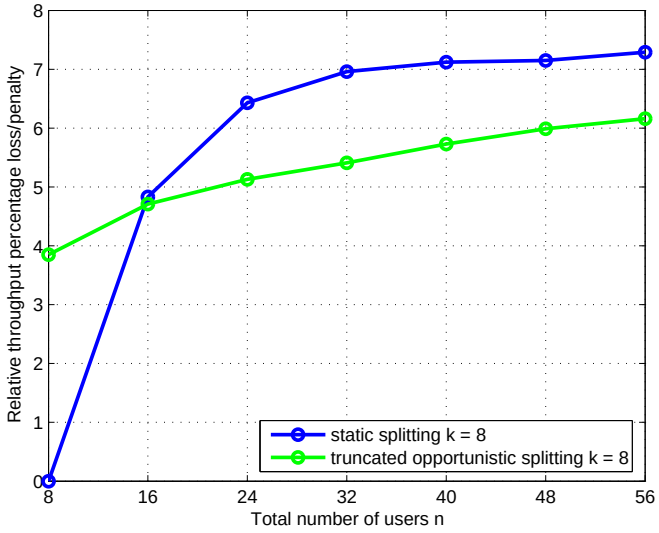



Figure 4.7: The relative throughput percentage loss due to static splitting and a truncated form of opportunistic splitting for $k = 8$ mini slots and an increasing number of users.

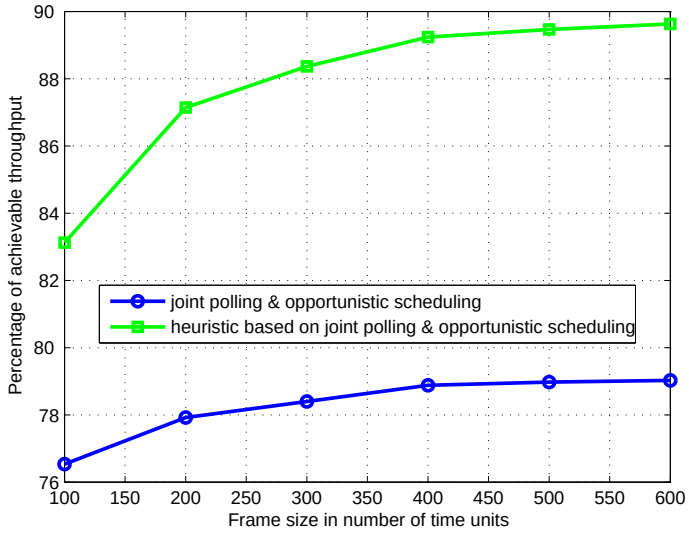



Figure 4.8: Percentage of achievable throughput realized by the JPOS scheme and it heuristic modification.

are able to achieve a fairly high fraction of the overall throughput possible, with the JPOS scheme achieving up to 79% and the heuristic modification achieving up to 89%.

## 4.5 Conclusion

In this chapter we presented a simple scheme to reduce feedback overheads under opportunistic scheduling in wireless networks. The scheme is novel in the sense that one can theoretically compute the required contention thresholds independent of users' distributions, making it applicable to real world scenarios. Unlike previous work our approach focused on finding a user with a high quantile, not necessarily the user with the highest quantile. The advantage of such an approach is verified by simulation results.

We also developed the insight that to reduce feedback in an opportunistic system where a mixture of real-time and best effort traffic is being served a combination of contention and polling based feedback approach is appropriate. We proposed two schemes based on this insight. Simulation results indicate that both schemes are able to exploit a large part of the available opportunism.

# Chapter 5

# Conclusion

In this dissertation we studied opportunistic downlink scheduling of users in centralized wireless networks. We considered measurement based scheduling of users with only best effort traffic in Chapter 2. There we showed that under fast fading the relative throughput penalty of a measurement based maximum quantile scheduling is fairly limited and grows only linearly with the number of users. By contrast, our experiments showed that other schemes might be more sensitive to measurement errors, and thus may not be able to exploit opportunism efficiently. Furthermore we developed the insight that suggests that a good way schedule users is to schedule a user that is experiencing the highest rate relative to its own distribution, i.e., maximum quantile scheduling. This conjecture is supported by our simulation results.

We studied providing quality of service guarantees via opportunistic scheduling in Chapter 3. The problem is challenging because in general the service seen by a user under opportunistic scheduling becomes dependent on the rate distributions of other users. Thus making it difficult to give any concrete QoS guarantee to a user. Additionally one would like to give the best effort users a high throughput and yet have a 'simple' resource allocation policy for the real-time users. In order to achieve these objectives, we proposed a token based scheduling scheme that exploits both inter and intra class opportunism and combined it with maximum quantile scheduling to decouple the service seen by a real-time user from other users' rate distributions, allowing us to give concrete rate guarantees to users. Furthermore we developed a stochastic lower bound on the service seen by a real-time user under the proposed scheme. The bound is significant, not only because it partially

captures the opportunism exploited, but also because it allows one to develop 'simple' resource allocation and call admission strategies. The advantage of using the proposed scheme is illustrated by simulation results which show that the proposed scheme can achieve 90% of the system capacity.

In Chapter 4 we first developed a simple static splitting scheme to reduce the amount of feedback required for opportunistic scheduling of best effort users. The scheme is combined with maximum quantile scheduling to find rate distribution independent optimum thresholds, thus allowing off-line calculations. The scheme aims to find a user that is currently experiencing a rate with a high quantile, but not necessarily the user that is experiencing the rate with the highest quantile. The advantage of such an approach is illustrated by our simulation results. Furthermore, we develop the insight that to reduce feedback while providing quality of service, one has to combine the contention based approach of soliciting feedback with dynamic polling. We propose two schemes based on this insight. Our simulation results show that the proposed schemes are able to meet users' quality of service requirements while exploiting a large fraction of opportunism present.

*Future Work.* There are several open problems remaining with respect to opportunistic scheduling in wireless networks. The focus of this dissertation is on scheduling at most one user at a time. It will be important to extend the ideas developed here to the scenario where more than one user can be scheduled at a time. An interesting question there might be to understand how the maximum quantile scheduler generalizes to that case.

The call admission policy developed Chapter 3 does not take into account the long term variations in users' channel capacity and the possibility of handover of calls from other cells. A more robust call admission criterion can be developed that takes into account these factors.

A further open problem in scheduling is finding the opportunistic scheduling strategy that minimizes delay (packet or file transfer) in a heterogenous sys-

tem supporting a dynamic load. (Some researchers have considered this problem when users have homogeneous channel capacity distributions in [36][42].) Finding such a strategy would be a fundamental advance as it would provide the key insight on the interactions between heterogeneity, opportunism and transfer delay required to extend optimal policies such as the shortest residual processing job first.

# Bibliography


[1] M. Andrews, K. Kumaran, K. Ramanan, A. Stolyar, R. Vijaykumar, and Phil Whiting. CDMA data QoS scheduling on the forward link with variable channel conditions. *Bell Laboratories Technical Report*, April 2000.

[2] P. Bender, P. Black, M. Grob, R. Padovani, N. Sindhushayana, and A. Viterbi. CDMA-HDR: A bandwidth-efficient high-speed wireless data service for nomadic users. *IEEE Communication Magazine*, pages 70 – 77, July 2000.

[3] D. Bertsekas and R. Gallagher. *Data Networks*. Prentice Hall, 1995.

[4] T. Bonald. A score-based opportunistic scheduler for fading radio channels. In *Proc. of European Wireless*, March 2004.

[5] T. Bonald and A. Proutiere. On stochastic bounds for monotonic processor sharing networks. *Queueing Systems*, 47:81 – 106, 2004.

[6] R. R. Boorstyn, A. Burchard, J. Liebeherr, and C. Oottamakorn. Statistical service assurances for traffic scheduling algorithms. *IEEE Journal on Selected Areas in Communications*, 18:2651 – 2664, December 2000.

[7] S. Borst. User-level performance of channel-aware scheduling in wireless data networks. In *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 1, pages 321 – 331, March-April 2003.

[8] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. The MIT Press, 1997.

[9] T.A. Cover and J.A. Thomas. *Elements of Information Theory.* John Wiley and Sons, 2002.

[10] G. C. Fox, R. D. Williams, and P. C. Messina. *Parallel Computing Works.* Morgan Kaufmann Publishers, 1994.

[11] D. Gesbert and M. Slim-Alouini. How much feedback is multi-user diversity really worth? In *Proc. IEEE Int. Conf. on Commun.*, pages 234 – 238, June 2004.

[12] Andrea Goldsmith. *Wireless Communications.* Cambridge University Press, 2005.

[13] A. Jalali, R. Padovani, and R. Pankaj. Data throughput of CDMA-HDR a high efficiency-high data rate personal communication wireless system. In *Vehicular Technology Conference Proceedings, 2000. VTC 2000-Spring Tokyo*, volume 3, pages 1854 – 1858, May 2000.

[14] Z. Ji, Y. Yang, J. Zhou, M. Takai, and R. Bagrodia. Exploiting medium access diversity in rate adaptive wireless lans. In *Proc. 10th Annual International Conference on Mobile Computing and Networking*, pages 345 – 359, September 2004.

[15] E.W. Knightly and N. B. Shroff. Admission control for statistical QoS: Theory and practice. *IEEE Network Magazine*, 13:20 – 29, March 1999.

[16] R. Knopp and P. Humblet. Information capacity and power control in single cell multi-user communications. In *Proc. IEEE International Computer Conference*, volume 1, pages 331 – 335, June 1995.

[17] X. Liu. Opportunistic scheduling in wireless communication networks, Ph.D. thesis, Purdue University. 2002.

[18] X. Liu, E. K. P. Chong, and N. B. Shroff. Opportunistic transmission scheduling with resource-sharing constraints in wireless networks. *IEEE*

*Journal on Selected Areas in Communications*, 19:2053 – 2065, October 2001.

[19] X. Liu, E. K. P. Chong, and N. B. Shroff. A framework for opportunistic scheduling in wireless networks. *Computer Networks*, 41:451 – 474, March 2003.

[20] A. Marshall and I. Olkin. *Inequalities: Theory of Marjorization and Its Applications*. Academic Press, 1979.

[21] A. Muller and D. Stoyan. *Comparison Methods for Stochastic Models and Risks*. John Wiley and Sons, 2002.

[22] 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 Globecom*, pages 528 – 532, November 2003.

[23] D. Park, H. Seo, H. Kwon, and B. G. Lee. Wireless packet scheduling based on the cumulative distribution function of user transmission rates. *IEEE Transactions on Communications*, 53:1919 – 1929, November 2005.

[24] R. Prakash and V.V. Veeravalli. Centralized wireless data systems with user arrivals and departures – Part I: Analysis. *Submitted to the IEEE Transactions on Information Theory*, January 2004.

[25] R. Prakash and V.V. Veeravalli. Centralized wireless data systems with user arrivals and departures – Part II: Applications and extensions. *Submitted to the IEEE Transactions on Information Theory*, January 2004.

[26] X. Qin and R. Berry. Opportunistic splitting algorithms for wireless networks. In *INFOCOM 2004. Twenty-Third Annual Joint Conference of the IEEE Computer and Communications Societies*, March 2004.

[27] X. Qin and R. Berry. Opportunistic splitting algorithms for wireless networks with heterogeneous users. In *Proc. Conference on Information Sciences and Systems (CISS)*, March 2004.

[28] T. S. Rappaport. *Wireless Communications, Principles and Practice.* Pearson Education, 2002.

[29] S. M. Ross. *Stochastic Processes.* John Wiley and Sons, 1983.

[30] Shahab Sanayei, Aria Nosratinia, and Naofal Aldhahir. Opportunistic dynamic subchannel allocation in multiuser OFDM networks with limited feedback. In *IEEE Information Theory Workshop*, October 2004.

[31] S. Shakkottai, R. Srikant, and Alexander L. Stolyar. Pathwise optimality of the exponential scheduling rule for wireless channels. *Advances in Applied Probability*, 36:1021 – 1045, December 2004.

[32] S. Shakkottai and A. Stolyar. Scheduling algorithms for a mixture of real-time and non-real-time data in HDR. In *Proc. of the 17th International Teletraffic Congress (ITC-17), Salvador da Bahia, Brazil*, September 2001.

[33] S. Shakkottai and A. Stolyar. Scheduling for multiple flows sharing a time-varying channel: The Exponential rule. *American Mathematical Society Translations, Series 2, A volume in memory of F. Karpelevich, Yu. M. Suhov, Editor*, 207, 2002.

[34] P. Svedman, S. K. Wilson, L. Cimini, and B. Ottersten. A simplified feedback and scheduling scheme for OFDM. In *IEEE Vehicular Technology Conference*, May 2004.

[35] T. Tang and R. W. Heath. Opportunistic feedback for downlink multiuser diversity. *IEEE Communication Letters*, 9:948 – 950, October 2005.

[36] L. Tassiulas and A. Ephremides. Dynamic server allocation to parallel queues with randomly varying connectivity. *IEEE Transactions on Information Theory*, 39:466 – 478, March 1993.

[37] P. Viswanath, D. Tse, and R. Laroia. Opportunistic beamforming using dumb antennas. *IEEE Transactions on Information Theory*, 48:1277 – 1294, June 2002.

[38] C. Westphal. Quantized scheduling for radio networks with heterogeneous fading characteristics. In *IEEE International Conference on Wireless And Mobile Computing, Networking And Communications (WiMob)*, pages 1 – 8, August 2005.

[39] D. Wu. Providing quality-of-service guarantees in wireless networks, Ph.D. thesis, Carnegie Mellon University. August 2003.

[40] D. Wu and R. Negi. Effective capacity: A wireless link model for support of quality of service. *IEEE Transactions on Wireless Communications*, 2:630 – 643, July 2003.

[41] D. Wu and R. Negi. Downlink scheduling in a cellular network for quality-of-service assurance. *IEEE Transactions on Vehicular Technology*, 53:1547 – 1557, September 2004.

[42] E. M. Yeh and A. S. Cohen. Delay optimal rate allocation in multiaccess fading communications. In *Proc. Allerton Conference on Communication, Control, and Computing*, pages 140 – 149, September 2004.

# Vita

Shailesh Patil was born to Late Siddappa Patil and Nanda Patil on July 14 1979 at New Delhi, India. He did his undergraduate studies from Netaji Subhas Institute of Technology, New Delhi, India. After working for a period of one year, he joined University of Texas at Austin in August 2002 to pursue graduate studies.

Permanent address: C-2-C, Pocket 2, 151
Janak Puri, New Delhi
India - 110058

This dissertation was typeset with LaTeX[†] by the author.

[†]LaTeX is a document preparation system developed by Leslie Lamport as a special version of Donald Knuth's TeX Program.