Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Computer Engineering and Networks Laboratory
Communication System Group

Semester Thesis

Vulnerability Evaluation of Proportional Fairness Scheduling to Retransmissions Attacks

Spring Semester 2010

Martin Baumgartner 05-912-985 <martibau@ee.ethz.ch>

Advisor: Udi Ben-Porat
Professor: Prof. Bernhard Plattner

Zurich
15th March 2011

Abstract

Channel aware schedulers of modern wireless networks – such as the popular Proportional Fairness Scheduler (PFS) – improve throughput performance by exploiting channel fluctuations while maintaining fairness among the users. In order to simplify the analysis, PFS was introduced and vastly investigated in a model where frame losses do not occur, which is of course not the case in practical wireless networks. Recent studies focused on the efficiency of various implementations of PFS in a realistic model where frame losses can occur.

Ben-Porat et al. [1] proved the immunity and fairness of their proposed variations to PFS by analytical analysis. In this work we back up these claims by simulation results. In addition we investigate two other aspects of the system: a) We compare the efficiency and vulnerability of the proposed solutions when used with different error correcting methods. b) We measure the transmission latency and propose a new solution to improve latency without compromising fairness or immunity.

Contents

1 Introduction **1**

2 Model **1**

2.1 The Proportional Fairness Scheduler 2

2.2 Frame Loss Handling Mechanisms 3

2.3 Channel Rate Mapping 5

2.4 Error Correcting Methods 5

3 Retransmissions Attack **5**

3.1 Vulnerability of the Proportional Fairness Scheduler 6

4 First Solution - Transmissions Average **7**

5 Second Solution - Effective Average **9**

6 Third Solution - Effective Average with Final Update **10**

6.1 Latency Comparison 11

7 Summary **13**

A Determining Simulation Parameters **14**

A.1 Acronyms 14

A.2 Data Rate 14

A.3 Reporting Channel Condition 16

A.4 Frame Error Rate (FER) 16

A.4.1 ARQ / HARQ 18

A.5 Steady State 18

A.6 Common Parameters to all Simulations 18

1 Introduction

High-speed wireless networks are becoming increasingly common. Along with that, the strategy of *scheduling* the high-speed data - which is vital to the network performance - has become the subject of active research. The modern wireless networks standards allow new generation of channel aware schedulers. One of the most popular such schedulers is the *Proportional Fairness Scheduler* (PFS) [2] [3] which improves throughput performance by exploiting channel fluctuations while maintaining fairness among the users.

In [1] it was shown that PFS is vulnerable to a simple NACK attack (see next section) in an environment where frame losses[1] occur. A modification of PFS was proposed and shown to be invulnerable to the attack while loosing the fairness property of PFS. *Effective Average* was introduced and shown to maintain *Proportional Fairness* as well as being resistant to the NACK attack. Finally, *Initial Effective Average* was proposed for PFS-FR (see subsection 2.2) that cures the vulnerability and maintains fairness of the original PFS. These approaches are modifications of PFS and are evaluated analytically in a model where each user's channel condition does not change over time, except for Effective Average.

In this work, we study[2] the PFS modifications proposed by [1] in a model where the user's channel condition change. An approach is introduced that increases average performance (better average latency[3]).

In Section 2 we describe the model used in this thesis.

In Section 3 we show that the Proportional Fairness Scheduler is also vulnerable when user's channel condition change.

In Section 4 we analyze the scheduler proposed in [1] that is immune to repeatedly reported NACK's in the general (non-CRA) model.

In Section 5 we show that this scheduler also maintains fairness among the users in the general model when some users repeatedly report NACK's.

Finally, in Section 6 we propose a fair and immune for practical use as the one presented in Section 5 but with better latency.

2 Model

In this section we describe the general *Proportional Fairness Scheduler* (PFS) and the different environments in which it is analyzed in this work.

[1]Some frames are not received successfully.
[2]With simulations
[3]The latency in context of scheduling is explained in section 4.

The model used in this thesis consists of a base-station that schedules down-link data to a fixed number of users. The base-station's objective is to deliver data in an efficient and fair manner.

2.1 The Proportional Fairness Scheduler (PFS)

As in [1] the wireless communication system discussed in this work consists of a base-station which serves its users with down-link data. The time is divided into time slots and the base-station is able to send data to exactly one user in each time slot (TDMA). The role of the scheduler (of the base-station) is to decide which of the users will be the next to receive data. PFS does that by assigning priority values to all users for every time slot, then the user with the highest priority is scheduled for transmission. Denote user i with U_i and his priority value for the scheduling decision of time t by $V_i(t)$. Then

$$V_i(t) = R_i(t)/A_i(t), \tag{1}$$

where $R_i(t)$ is the bit rate (measured in bits per time slot *bits/slot*) in which the system sends data to U_i if he is assigned time slot t. The value of $R_i(t)$ is decided by the scheduler according to the channel condition of the user as expressed by the signal-to-noise ratio (SNR) reported by the user for every time slot. $A_i(t)$ is the throughput average of U_i until time slot t (not including) measured in *bits/slot*. The throughput average is updated after every time slot. Each user *pays* a *price* for each bit received. This *price* for getting a transmission in time slot t results in a decreased probability to win a time slot in the future. A simple method of updating the average (*price*) is:

$$A_i(t+1) = \frac{t-1}{t}A_i(t) + \frac{1}{t}R_i(t)1_i^{rcv}(t). \tag{2}$$

where $1_i^{rcv}(t) = 1$ if U_i received data on slot t, $1_i^{rcv}(t) = 0$ otherwise. Another method of averaging is "discount averaging" which gives less weight to the data received in the past:

$$A_i(t+1) = (1-\epsilon)A_i(t) + \epsilon R_i(t)1_i^{rcv}(t), \tag{3}$$

In this work it is referred to these averaging methods[4] (Eq. 2 or 3) as *Admitted Average* since in the *Loss Model* (where frame loss are possible) $1_i^{rcv}(t) = 1$ only if the user admitted to receive the transmission successfully by reporting ACK or when the retransmission limit is reached. This is done in order to differ it from new averaging methods suggested later in this work. According to the above equations, the throughput average of the user

[4]Simulations showed that the results were independent of which of these is used.

assigned for transmission is not decreased like the averages of all the other users. It means that in the next time slot it is harder for him to obtain the highest priority value (Eq. 1). Therefore, throughout the work it is referred to the value multiplied with $1_i^{rcv}(t)$ (in both methods) as the "price" that the user "pays" for "winning" the time slot. The higher the price is, the higher his updated throughput average will be and this lower his potential priority value in the following time slots (Eq. 1). [1]

Note that in practice it is hard to differ between both averaging methods (Eq. 2 and Eq. 3) especially when the common recommended value $\epsilon = 0.001$ is used [2]. In order to avoid floating point errors[5], the simulations for this thesis are computed with formulas deducted from Eq. 3. In addition, it was proved in [4] that regardless of the method used, PFS complies with the *Proportional Fairness* criterion given by Kelly [5], [6] which maximizes the utility function θ given in Eq. 4:

$$\theta(N) = \sum_{i=1}^{N} log(A_i), \tag{4}$$

where N is the number of users in the system and A_i is the throughput average of U_i under the steady state. As in [7] this thesis uses the *normalized utility function* to measure the *efficiency*:

$$Eff(R) = \frac{\sum\limits_{i=1}^{|R|} log(\bar{A}_i)}{|R|} \tag{5}$$

where R is the set of regular users and $\bar{A}_i$ is the average number of bits per time slot user i received successfully[6]. Only time slots in steady state are considered.

For the visualization of the efficiency the *efficiency loss* is used:

$$Eff\_Loss(R, N) = 1 - \frac{Eff(R)}{Eff(N)} \tag{6}$$

where R is the number of regular users and N the total number of users. Note that $Eff(N)$ is the efficiency of a system[7] with only regular users.

2.2 Frame Loss Handling Mechanisms

PFS as described in the previous section was introduced and defined in the *Lossless Model*. When implementing PFS for the *Loss Model*, that is, an environment where frame losses can occur, two major issues have to addressed: 1) *Effective Rate* evaluation; 2) Frame loss handling.

[5] $\frac{1}{t}$ when $t \gg 1$

[6] $\bar{A}_i$ is equal to A_i as defined in Eq. 2 but without the time slots prior to steady state.

[7] simulation run

Effective Rate Evaluation In PFS for the *Lossless Model* the numerator of the priority value is $R_i(t)$ - the data rate in which the SNR reported by the user was mapped to. If frame losses can occur then the rate R_i in which we send data to the user does not represent the actual bit-rate the user receives successfully. Therefore, in the *Loss Model*, $R_i(t)$ at the numerator of the priority value (Eq. 1) is replaced with an *Effective Rate* value, denoted by $R_i^e(t)$, which is the rate that the user is expected receive successfully (also measured with $bits/slot$). The common way to calculate the effective rate is by $R_i^e(t) = G_i(t)R_i(t)$ where $G_i(t)$ is the probability of successful transmission when sending data in rate $R_i(t)$ given the SNR value he reported for that time slot. Eq. 7 concludes the calculation of the priority value in the *Loss Model*:

$$V_i(t) = \frac{R_i^e(t)}{A_i(t)} = \frac{G_i(t)R_i(t)}{A_i(t)}. \tag{7}$$

For example, if the actual rate that will be transmitted to the user is $R_i(t) = 100 bits/slot$ and the probability that the user will receive it correctly is $G_i(t) = 0.8$, then the effective rate is $R_i^e(t) = 80 bits/slot$. More about the calculation of the effective rate for PFS in the *Loss Model* can be found in [7].

Frame Loss Handling The second issue that has to be addressed is how to handle cases of lost frames, that is, how to handle a frame waiting to be retransmitted after its previous transmission failed (the user replied with NACK). There are two approaches to address it:

- *Proportional Fairness Scheduler with Fast Retransmission (PFS-FR)* - The frame will be immediately retransmitted to the user in the next time slot whether he obtains the highest priority value or not.
- *Proportional Fairness Scheduler with Slow Retransmission (PFS-SR)* - The frame will be retransmitted only in the earliest time slot the user obtains the highest priority value.

Note that (for both PFS-SR and PFS-FR) the throughput average (A_i) of a user is the *Admitted Average* as defined in 2.1 since it represents the average of the actual data he received also in the *Loss Model*. Formally, Eq. 3 (or 2) is used as is, and $1_i^{rcv}(t) = 1$ only if the user received the frame on time t, that is, his feedback for the frame is ACK. As mentioned in Section 2.1, this throughput average method is used as *Admitted Average*. Note that in order to prevent an abuse of the system, the system limits the number of possible retransmissions to a fixed value - L_{max}. If the limit is reached then the user's throughput average is updated as if he received the frame successfully even if he reported NACK.

2.3 Channel Rate Mapping

The reported channel condition of each user remains the same while a he's waiting for a retransmission[8]. All users that are not waiting for a pending transmission get assigned a Rayleigh distributed channel condition (see Appendix) in each time slot by the simulation program [2].

2.4 Error Correcting Methods

Error correcting methods are used to reliably send data on an unreliable channel. While for analytical approaches it is easiest not to consider any error correcting methods these are widely used in real-world applications. In this thesis we analyze the impact of a popular error correcting method and make simulations with

- *Automatic Repeat Request (ARQ)* - The frame is decoded from a single transmission. In case of an unsuccessful reconstruction a retransmission is ordered while the previous transmissions are not considered for decoding the message.
- *Hybrid Automatic Repeat Request (HARQ) with Chase Combining* - Once a transmission failed the user collects the information from that transmission. Since all retransmission frames sent are identical to the initial frame for that message the probability to decode a message correctly increases when all signals from all collected transmissions for that message are combined.

Both methods have in common that the frame is retransmitted until it is received successfully[9] or the maximum number of retransmissions (L_{max})is reached.

3 Retransmissions Attack

In [1] it was shown that the Proportional Fairness Scheduler (PFS) is vulnerable in a *loss model* to a *Retransmission Attack* in which a malicious user reports NACK even when he receives the frame correctly [1].

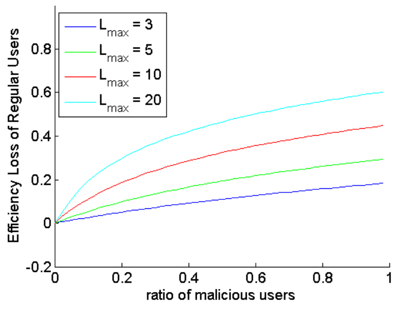
The only goal of such (malicious) user is to decrease the time share of regular users. This kind of behavior results in an increased time share for the

[8]This is realistic especially when the retransmissions take place few time slots after the initial transmission.

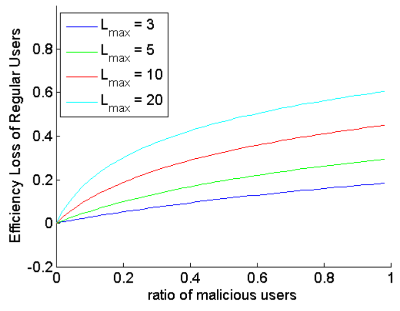
[9]A transmission consists of the message and its CRC. The user reports ACK if the calculated CRC of the received message equals the received CRC.

malicious user(s) on the expense of the regular users served by the scheduler. The vulnerability of the scheduler to such attacks is explained in the following way: In the *Loss Model*, if U_i reported NACK, then $1_i^{rcv}(t) = 0$ and his average decreases exactly as for the other users $\forall_{j \neq i}.\ \frac{A_i(t+1)}{A_j(t+1)} = \frac{A_i(t)}{A_j(t)}$. That is, the fact that the scheduler has put efforts in transmitting data to U_i at time t does not count against him in future time slots (unlike the *Lossless Model* where if a frame transmitted to a user it means he received it). A malicious user can abuse this insensitivity property of the *Admitted Average* and report NACK in every opportunity he has[10].

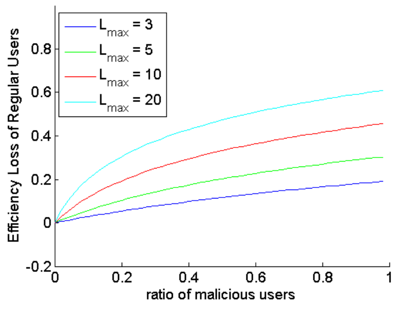
3.1 Vulnerability of the Proportional Fairness Scheduler



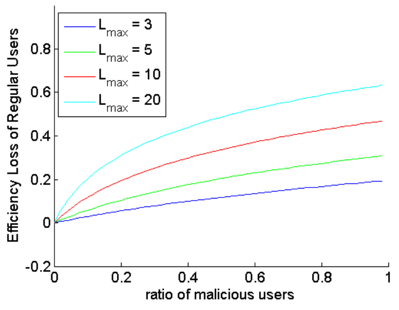
(a) PFS-SR with ARQ



(b) PFS-FR with ARQ



(c) PFS-SR with HARQ



(d) PFS-FR with HARQ

Figure 1: Simulation results of efficiency loss under steady state for the regular users. This graph resulted from a simulation run with 50 users and the original PFS.

To analyze the PFS we did run simulations with 50 users and analyzed the efficiency of the regular users in steady state[11] for both PFS-SR and PFS-

[10] As long as the retransmissions limit (L_{max}) is not reached.

[11] It is assumed that the steady state is reached, when the priority values of the users

FR. Different L_{max} values and different ratios of regular users to malicious users were tested.

Figure 1 shows the vulnerability of PFS to a retransmission attack. Each point correspondents to one simulation run. There are 50 users in the system and the X-axis indicates how many of them are malicious. The Y-axis (*efficiency loss*) is calculated as in Eq. 6. One can see that the vulnerability of PFS-FR and PFS-SR as well as *with ARQ* and *with HARQ* are not very different from each other. The efficiency of the regular users drops as the number of malicious users increases. As the L_{max} is increased, each malicious user gets more time slots before he is *forced to ACK*. This makes an attack more powerful. More malicious users or a higher L_{max} decreases the efficiency of regular users in PFS.

Other simulations showed that the efficiency loss is not affected by the total number of users in the system but only by the percentage of malicious users. This is why we chose the percentage of malicious users as x-axis of the graphs.

Due to the attack the system sees more NACK's than with regular users only. This leads to a lower throughput for all users because PFS maintains fairness also when there are more NACK's. In steady state under CRA it was shown in [1] that the proportion of the priority values of every pair of users remains the same when no user received a data frame. It can be proved that under CRA the priority value is proportional to the throughput average of all users (also the non-malicious ones) decreases with every NACK.

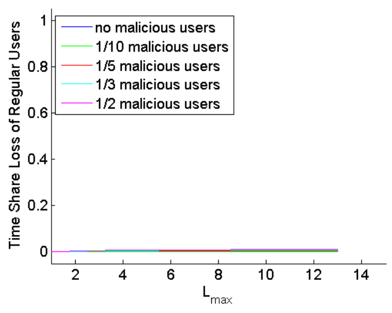
In the general model however the priority values change as the SNR values change. Simulations showed that the average priority value of every user converges to the same level independently of the user's behavior because PFS is fair [1].

4 First Solution - Transmissions Average

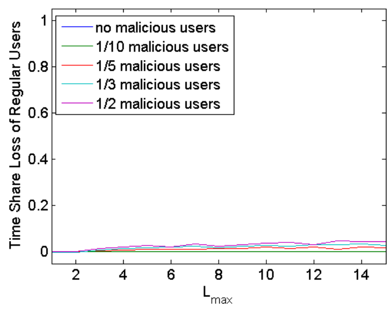
In [1] it was shown that *Transmission Average* cures the vulnerability of PFS it distorts the fairness of the scheduler. In this section we discuss our simulations of a solution which is invulnerable to a retransmission attack and was proposed by [1].

The problem in Section 3.1 was created because the user "pays" according to the amount of resources the system *expects* him to use, and not according to what he actually uses. Therefore, the motivation behind this solution is to "charge" all the users with the efforts the system invests in them instead of

$V(t)$ remain at the same level over a long period of time. Simulations showed that this is usually the case after 1000 time slots for simulations with 50 users. All graphs shown in this thesis assume that time slots after the 5000th are in steady state.



(a) PFS-SR with ARQ and Transmission Average



(b) PFS-SR with HARQ and Transmission Average

Figure 2: PFS-FR with Transmission Average: Time share loss of regular users in steady state. Note that the number of users in the simulation to produce this graph is 50.

charging them with what they admit they got. This way, the ACK/NACK feedbacks of the users cannot influence the scheduling decisions (through the average throughput). In this solution, after every time slot in which the system transmitted $R_i(t)$ to the user, $A_i(t)$ is updated with $R_i(t)$ regardless of his feedback for this transmission (ACK/NACK). Formally, Eq. 2 is replaced with

$$A_i(t+1) = \frac{t-1}{t}A_i(t) + \frac{1}{t}R_i(t)1_i^{snd}(t), \tag{8}$$

where $1_i^{snd}(t) = 1 \iff V_i(t) = max\{V_j(t)\}_{j=1}^{N}$, that is $1_i^{snd}(t) = 1$ if the scheduler sent U_i a frame at time slot t regardless of the ACK/NACK feedback (otherwise $1_i^{snd}(t) = 0$). This averaging method is known as *Transmissions Average* since it measures the average of the transmissions instead of what the user received.

It was proved analytically in [1] that a user cannot gain anything by reporting fake NACK's in PFS[12] with transmission average.

The graphs shown in Figure 2 are produced using simulations with 50 users and different L_{max} (X-axis). The Y-axis is calculated as

$$\text{Time Share Loss of Regular Users} = 1 - \frac{\sum_{i=1}^{|R|} ts_i}{|R|} \tag{9}$$

where R is the set of regular users and ts_i the time share of a (regular) user i in steady state.

[12]Both PFS-SR and PFS-FR

Figure 2 shows that as expected whether the retransmission limit L_{max} not the number of malicious users do have an influence on the time share of regular users.

Remark 1 *The latency is the time that passes*[13] *between the first transmission of a frame until it is received successfully or the number of transmissions is L_{max}*[14]. *What is presented in this remark holds only for PFS-SR as the latency of PFS-FR is always minimal due to immediate retransmission.*

If user U_i is assigned the time slot t, A_i is increased independently of the success of the transmission $\Rightarrow V_i(t+1) < V_i(t)$[15]. The chance for user U_i to win the next time slot decreases with every unsuccessful transmission. The more users there are in the system the more likely it is that several of them are waiting for a retransmission at the same time which increases the average latency. A graph showing this phenomena will be presented in Section 6.1.

5 Second Solution - Effective Average

In [1] it was shown that *Effective Average* is not vulnerable to retransmission attacks and maintains fairness among the users under general rate conditions. Simulations of this modification to PFS were made and will be discussed in this section

It was shown in [1] that the solution proposed in the previous section is only fair as long as all users have the same frame error rate[16] which is rarely the case in reality.

In this solution the throughput average $A_i(t)$ is the expected throughput a user U_i should have gotten so far, given the efforts the system has put into him. Formally, Eq. 2 is replaced with

$$A_i(t+1) = \frac{t-1}{t}A_i(t) + \frac{1}{t}R_i^e(t)1_i^{snd}(t), \tag{10}$$

where $1_i^{snd}(t)$ is as defined in Section 4 and $R_i^e(t)$ in Section 2.2. Besides the calculation of A_i the simulation settings for both Figure 3 and Figure 1 are the same. The curves show that PFS with *Effective Average* other than

[13]number of time slots

[14]In this case the transmission is abandoned and the scheduler assumes that the frame was received.

[15]It can be proved that this is true assuming that the reported channel condition of user i does not change while he's waiting for retransmission.

[16]The frame error rate is the chance of a transmission to fail. It is defined by the modulation of the transmission and the channel condition of the user.

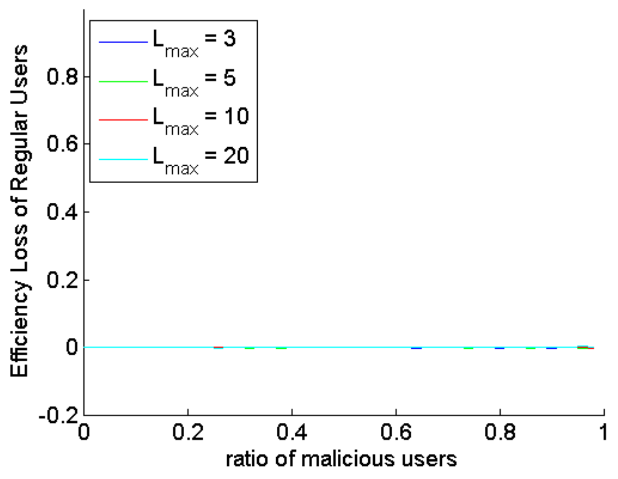


Figure 3: Simulation results of efficiency loss for the regular users under steady state for PFS-SR with ARQ. This graph resulted from simulation runs with 50 users, different values for L_{max} and the PFS with Effective Average.

Admitted Average is immune to the retransmission attack even with general channel conditions for both PFS-SR and PFS-FR as expected.

PFS-SR with *Effective Average* is exposed to a high average latency for the same reason as with *Transmission Average*.

6 Third Solution - Effective Average with Final Update

In the previous section *Effective Average* was studied which is is not vulnerable and maintains fairness among the users but has a high average latency. In this section we propose a solution for PFS-SR that performs nearly as good as PFS with *Effective Average* but with a lower average latency.

While the immediate increase of A_i after each transmission makes both methods - *Transmission Average* and *Effective Average* - immune to retransmission attacks it leads to a higher latency in PFS-SR. As retransmission in PFS-FR happens immediately the problem with increased latency does not apply to PFS-FR. We try to combine the advantage of both methods:

- A user with an outstanding channel condition will get a packet transmitted to it independently of other users waiting for a retransmission (advantage of PFS-SR)

- Once a user is waiting for retransmission it is likely that the next time slot is assigned to it (advantage of PFS-FR)

We propose to increase A_i for a user only after a transmission ended but with the same amount as with *Effective Average*:

$$A_i(t+1) = \frac{t-1}{t}A_i(t) + \frac{1}{t}R_i^{e,sum}(t)1_i^{rcv}(t), \tag{11}$$

where $1_i^{rcv}(t)$ is defined as in 2.1 and $R_i^{e,sum}(t)$ equals the sum of all the effective rates of the time slots needed to send this packet. Eq. 11 is only an approximation of the original Eq. 10 as this example shows: A user U_i gets a packet transmitted in two consecutive time slots t (NACK) and $t+1$ (ACK). According to Eq. 10 $A_i^e(t+2) = \frac{t-1}{t+1} * A_i(t) + (\frac{1}{t+1} + \frac{1}{t}) * R_i^e(t)$ while with Eq. 11 $A_i^{e,sum}(t+2) = \frac{t-1}{t+1} * A_i(t) + \frac{2}{t} * R_i^e(t) \overset{t \gg 1}{=} A_i^e(t+2)$.

Simulations with the same settings as for Figure 3 in Section 5 except for time A_i is updated showed that similar results in time share loss.

Note that with this solution malicious users can take over consecutive time slots as with PFS-FR. This results in a decreased efficiency. Even if the latency is reduced the average amount of time that passes since a packet arrived to the scheduler until it was received successfully by the user does not change.

6.1 Latency Comparison

The curves in Figure 4 are produced by several simulation runs with different numbers of regular users[17]. The average latency of a user is the sum of all time slots that passed between a first transmission of a packet to that user until the packet is received successfully. The average latency of regular users (Y-axis) shown in the graph is the average over all user's average latency.

As the number of users is low (N<5) there is no difference in latency between *Effective Average* and *Effective Average with Final Update.* When the number of users gets large (N>30) the latency of *Effective Average* increases linearly with the number of users where the latency of *Effective Average with Final Update* increases only slightly. *Effective Average with Final Update* does a massive improvement to the average latency when there are many users in the system.

[17]There are no malicious users in the system.

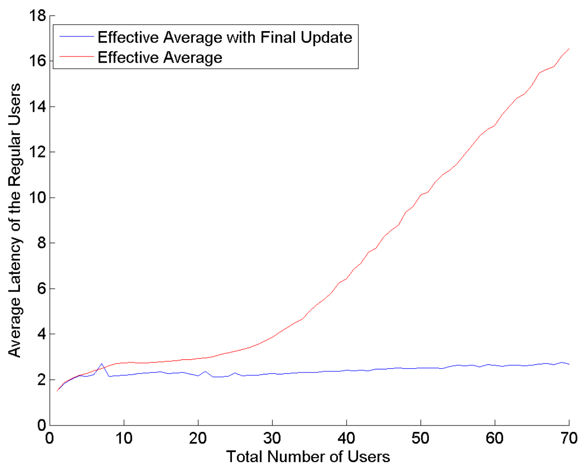


Figure 4: This curves show the average latency of transmissions in steady state for PFS-SR with ARQ for *Effective Average* and *Effective Average with Final Update*. There are no malicious users in the system $(N = R)$.

7 Summary

We showed that in practical networks where frame loss are considered "standard" Proportional Fairness is vulnerable to malicious attacks. We aimed at devising a policy that is immune to such attacks while maintaining fairness. We analyzed the approaches from [1] in an environment with changing channel conditions and showed that the approaches are applicable here as well. We proposed a scheduler for PFS-SR which maintains fairness, is immune to retransmissions attacks and decreases the average latency of a data packet.

Acknowledgment

Thanks to Udi Ben-Porat which supported me with many helpful inputs.

A Determining Simulation Parameters

This chapter describes decisions that have to be made concerning the behavior of a user and its interaction with the system.

A.1 Acronyms

The appendix uses the following acronyms:

a_i throughput average of user i
ARQ Automatic Repeat reQuest
FER frame error rate
HARQ Hybrid ARQ with chase combining
i a specific user i among all users
k a specific modulation and coding scheme (MCS)
L number of time slots assigned to a user-frame pair
L_{max} maximum number the same frame is allowed to be transmitted
MCS modulation and coding scheme
N total number of users (users) in the system
PFS Proportional Fairness Scheduling
PFS-FR Proportional Fairness Scheduling (PFS) with Fast Retransmission

R data rate - Note that in the thesis R usually stands for the set of regular users in the system.
R_{eff} effective rate
R_i data rate of a user i
R_k data rate of MCS$_k$
S set of values a user can report as its *SNR*
SNR signal-to-noise ratio
system base station
user mobile device

A.2 Data Rate

The system (base station) chooses the frame size to be transmitted in the time available. In the following one uses only the term *data rate - Note that in the thesis R usually stands for the set of regular users in the system. (R)* to describe the size of the frame[18]. The data rate cannot be chosen arbitrarily as each data rate of a user i (R_i) has its own modulation and encoding. The data rates used in this paper origin from the table described in section A.4.

[18]The length of a time slot is fixed $\rightarrow R \propto$ frame size

For every signal-to-noise ratio (*SNR*) value (see A.3) one can calculate the MCS that potentially gives the highest R as follows[19]:

$$R(SNR) = R_k \qquad k : R_{eff}(MCS_j, SNR) = max_l(R_{eff}(MCS_l, SNR)) \\ l \in \{1, \ldots, 13\},\ R_{eff} := R * (1 - FER) \tag{12}$$

This results in one data rate per section of *SNR* values: 1.

Modulation Number	Data Rate in Mbps (R_i)	Min SNR in dB (**S**)	Max SNR in dB	Modulation Number k in Simulator (MCS)
1	0.48	-7.3	-5.0	1
2	0.72	-4.9	-2.1	2
3	1.44	-2.0	1.0	3
4	2.88	1.1	4.1	4
5	0.72	-	-	-
6	0.72	-	-	-
7	0.72	-	-	-
8[20]	4.32	$-\infty$	-7.4	-
8	4.32	4.2	6.4	5
9	5.76	6.5	10.3	6
10	8.64	10.4	14.2	7
11	8.64	-	-	-
12	11.52	14.3	16.4	8
13	12.96	16.5	18.3[21]	9

Table 1: SNR mapping
SNR mapping: not all modulations are used in simulation because some (5, 6, 7, 11) are not competitive in our simulation environment.

[1] This section of *SNR* values is not taken into account due to very high frame error rate (FER).
[2] All values above 16.5dB are mapped to this modulation. 18.3dB is chosen as upper value to get a reasonable modulation distribution (see A.3).

[19]Note that this is only correct for Automatic Repeat reQuest (ARQ). The effective rate (R_{eff}) for Hybrid ARQ with chase combining (HARQ) is computed differently. For simplicity this thesis uses the same values for both simulation types ARQ and HARQ. In the program however there is an option to simulate HARQ with its corresponding values.

A.3 Reporting Channel Condition

The user cannot report it's precise channel condition[22] but has to report a natural number instead [7].

In order to make the simulation results better understandable one uses the same number of MCS as there are data rates[23], see A.2. To every MCS$_k$ there is a SNR_k which the system assumes that it's the user's channel condition. The user reports the modulation and coding scheme (MCS) that matches its actual SNR_i best[24]. The corresponding SNR values are determined from the table described in section A.4:

$$\mathbf{S} = \{-7.3, -4.9, -2, 1.1, 4.2, 6.5, 10.4, 14.3, 16.5\}^{25} \tag{13}$$

The SNR in the simulations are Rayleigh distributed [8]. The σ of the Rayleigh function is calculated as follows:

$$\begin{aligned} \sigma &= average(SNR_{min}, SNR_{max})\text{dB} && (14)\\ &= \frac{-7.3 + 18.3}{2}\text{dB} && (15)\\ &= 5.5\text{dB} && (16)\\ &= 1.883649089489800 && (17) \end{aligned}$$

This leads to the following distribution of reported MCS:

SNR_i values do change with every time slot. The simulation program assumes that consecutive channel conditions are independent to each other, which is not the case in reality.

A.4 Frame Error Rate (FER)

$$FER_i(t) = 1 - G_i(t) \tag{18}$$

As described in 2.2 one needs to know the probability that the frame will not be received successfully in order to simulate whether a transmission was successful: the frame error rate (FER). It's influenced by various factors:

- channel condition (SNR)
- data rate of MCS$_k$ (R_k)

[22]The channel condition is expressed as signal-to-noise ratio (SNR)

[23]This results in a system where the user wishes for a R for the next time slot

[24]Example: the user has a SNR of 1.4dB. The nearest SNR in the set of values a user can report as its SNR ($\mathbf{S}$) is 1.1dB which corresponds to MCS 4. So the user reports MCS 4.

[25]For HARQ the values are
$\mathbf{S}_{\text{HARQ}} = \{-40, -20.2, -17.4, -14.4, -11.1, 0.2, 10.3, 14.1, 16.4\}$.

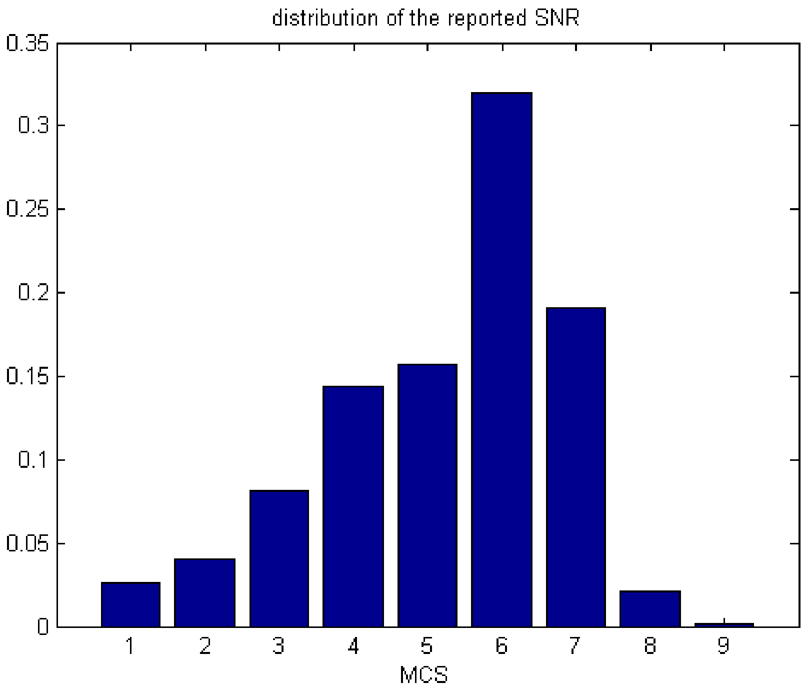


Figure 5: Due to the Rayleigh distribution not all MCS are reported equally often.

- error correcting mechanisms used

The thesis uses a FER table which provides FER values for pairs of SNR_k and R_k. There are only two error correcting mechanisms simulated: ARQ (Stop-and-Wait [9]) and HARQ [10]. To determine the FER in a HARQ transmission one sums up channel conditions[26] to get an *effective channel condition*, which can be looked up. The values are used by the system in order to determine the next user to be served and the R_k, as well as by the user[27] to determine whether the transmission was successful or not. Note that the used value is often not the same as the system knows only about the reported SNR_k, which is an approximation of the actual SNR_i.

The FER is higher the higher the R is and the lower the SNR is.

A.4.1 ARQ / HARQ

In Automatic Repeat reQuest (ARQ) no error correcting mechanism is used [9]. When a frame fails the user orders it again. In Hybrid ARQ with chase combining (HARQ) the probability for a transmission to fail decreases with every transmission of the same frame as the user keeps track of the failed transmissions [10]. In simulations using HARQ the FER is in addition to the data rate and the SNR also dependent of the number of previous transmissions of the same frame (L).

A.5 Steady State

At initialization, the throughput average (and A_i) of all users are zero[28]. That is why the priority decreases in the first part of every simulation until a level of priority-stabilization is reached.

One assumes that the system is stable, if the average priority value remains at the same level for some period of time. Depending on the input parameters stabilization is reached within the first < 3000 time slots as simulations of our own have shown. For the simulation graphs the time slots from the 5000th on are considered to be after reaching steady state[29].

A.6 Common Parameters to all Simulations

All simulations shown throughout this thesis are based on the following scenario, if not explicitly specified otherwise:

[26] Chase combining

[27] In reality FER values are used by the base station (system) only!

[28] $1 >> a_i > 0 \forall i \in \{1, \ldots, N$

[29] All graphs shown in this thesis take into account $5000 \leq t \leq 200000$

- 50 users in total (N= 50)
- The simulation runs for 200'000 time slots. The first 5000 time slots are not considered to produce the graphics shown in this thesis (see A.5)
- There are two types of users: regular[30] ones and malicious[31] ones.
- The throughput average is calculated on a *discount based method* (e.g. Eq. 2)
- The PFS with Fast Retransmission (PFS-FR) is used
- maximum number the same frame is allowed to be transmitted (L_{max})= 5.

[30]Regular users do behave as expected.

[31]Malicious users do never report ACK: NACK-attack. By always requesting the sent packet again a user seeks to increase the number of time slot assigned to itself and thereby decreasing the other users share of the channel.

References

[1] U. Ben-Porat, A. Bremler-Barr, H. Levy, and B. Plattner, “On the vulnerability of the proportional fairness scheduler to retransmission attacks,” infocom 2011.

[2] R. P. A. Jalali and R. Pankaj, “Data throughput of cdma-hdr: A high efficiency high data rate personal communication wireless system,” in *Proceedings of the IEEE Vehicular Technology Conference*, 2000.

[3] S. Borst, “User-level performance of channel-aware scheduling algorithms in wireless data networks,” *IEEE/ACM Trans. Netw.*, vol. 13, no. 3, pp. 636–647, 2005.

[4] P. A. W. Harold J. Kushner, “Asymptotic properties of proportional-fair sharing algorithms,” *Proc. 40th Annual Allerton Conf.*, pp. 1051–1059.

[5] F. Kelly, “Charging and rate control for elastic traffic,” *European Transactions on Telecommunications*, vol. 8, pp. 33–37, 1997.

[6] H. J. Kushner and P. A. Whiting, “Convergence of proportional-fair sharing algorithms under general conditions,” *IEEE Trans. Wireless Commun*, vol. 3, pp. 1250–1259, 2003.

[7] P. D. Systems, H. Zheng, H. Viswanathan, and S. Member, “Ieee transactions on wireless communications, vol. 4, no. 2, march 2005 495 optimizing the arq performance in downlink,” *IEEE Trans. Wireless Commun*, vol. 4, pp. 495–506, 2005.

[8] A. J. Goldsmith, S. ghee Chua, and A. Member, “Variable-rate variable-power mqam for fading channels,” *IEEE Trans. Commun*, vol. 45, pp. 1218–1230, 1997.

[9] F. B. Schneider and F. B. Schneider, “Implementing fault-tolerant services using the state machine approach: A tutorial,” *ACM Computing Surveys*, vol. 22, pp. 299–319, 1990.

[10] D. Chase, “Code combining–a maximum-likelihood decoding approach for combining an arbitrary number of noisy packets,” *Communications, IEEE Transactions on*, vol. 33, no. 5, pp. 385 – 393, may. 1985.

Statement regarding plagiarmism when submitting written work at ETH Zurich

By signing this statement, I affirm that I have read the information notice on plagiarism, independently produced this paper, and adhered to the general practice of source citation in this subject-area.

Information notice on plagiarism:
http://www.ethz.ch/students/semester/plagiarism_s_en.pdf

Supervisor	____________	________________________
Student	____________	________________________
Place and date	____________	Signature ________________

26.04.10 / NB