

Quantized Scheduling for Radio Networks with Heterogenous Fading Characteristics

Cédric Westphal

Abstract—As the 3G cdma2000[®] networks with High Data Rates get deployed, it is interesting to study the impact of the scheduling algorithms used in these networks, and to improve over their performance. In this paper, we introduce a quantized scheduling that takes into account the fact that mobile nodes and access points can communicate only through a finite set of rates or power vectors. This approach allows also to improve over the traditional scheduling algorithm used in the cdma2000 networks, proportional fairness, in terms of fairness in heterogenous channels (that is channels where the fading is governed by different distributions for different nodes) by mapping all channels to a vector of i.i.d. random variables from which to base the scheduling decision. There is no performance degradation for using this algorithm in Rayleigh fading channels, but improved fairness in the heterogenous fading case.

Index Terms—Scheduling, proportional fairness, uplink, downlink, forward and reverse channels, CDMA HDR.

I. INTRODUCTION

NE of the drivers for the next generation of cellular networks is the expected need for higher bandwidth levels. To achieve this, scheduling algorithms which take into account the link quality are behind standardized and deployed. The scheduling algorithm allocates the channel from one base station to one of several contending nodes based on the current conditions of the channel. The fading characteristic from one node to the base station can exhibit a different rate distribution than that of another node. This we denote as the heterogenous case. Conversely, the homogenous case would be if all nodes' fading follow the same pattern, for instance, Rayleigh fading.

Among the so-called opportunistic scheduling algorithms, Proportional Fairness (PF) is the algorithm in CDMA High Data Rate (HDR) networks [1], [2], [3]. CDMA HDR networks are being standardized through 3GPP2 [4]. PF is designed to schedule the

node whose link quality is the best when normalized with its average throughput. This is where its name comes from: its definition of *fairness* is that it achieves a rate *proportional* to the mean channel rate. However, the algorithm does not achieve this goal when the channels to different mobile devices are heterogenous or when the device is rate limited [5], [6].

Several improvements have been made to compensate these shortcomings. A score-based (SB) scheduler is proposed in [6]. However, this scheduler is either sensitive to correlation in the packet rates, or requires to store a large amount of data at the scheduler (namely, an extended history of the rate requests for each node). We propose a different scheme that does not suffer from correlation between packets, and as such, does not require to maintain any more state than a sliding window average for a few quantities for each mobile node.

PF works as follows: for channels (or mobile nodes) $j = 1, \dots, N$ requesting rate r_j at the scheduling decision time epoch t , the PF scheduler chooses nodes η such that:

$$\eta = \arg \max_{j \in \{1, \dots, N\}} \left(\frac{r_j}{R_j} \right) \text{ where:}$$

$$R_j(t+1) = (1 - \alpha)R_j(t) + \alpha \mathbf{1}_{\{\eta=j\}} r_j \quad \text{and } 0 < \alpha < 1 \quad (1)$$

The SB considers its scheduling decision by ranking the rate $r_j(t)$ against the W previous values $r_j(t-1), \dots, r_j(t-W)$, thus yielding an ordering *score* $w_j \in \{1, \dots, W\}$. It schedules the channel for which: $\eta = \arg \max_j w_j$ with the appropriate tie breaker.

II. PROBLEM MOTIVATION

Both the PF and SB scheduling policies base their scheduling decisions on an internal reference point: as such, the decision for each node are consistent with the previous decisions for the same node, but

C. Westphal is with the Communication System Laboratory, Nokia Research Center, Mountain View, California. E-mail: cedric.westphal@nokia.com

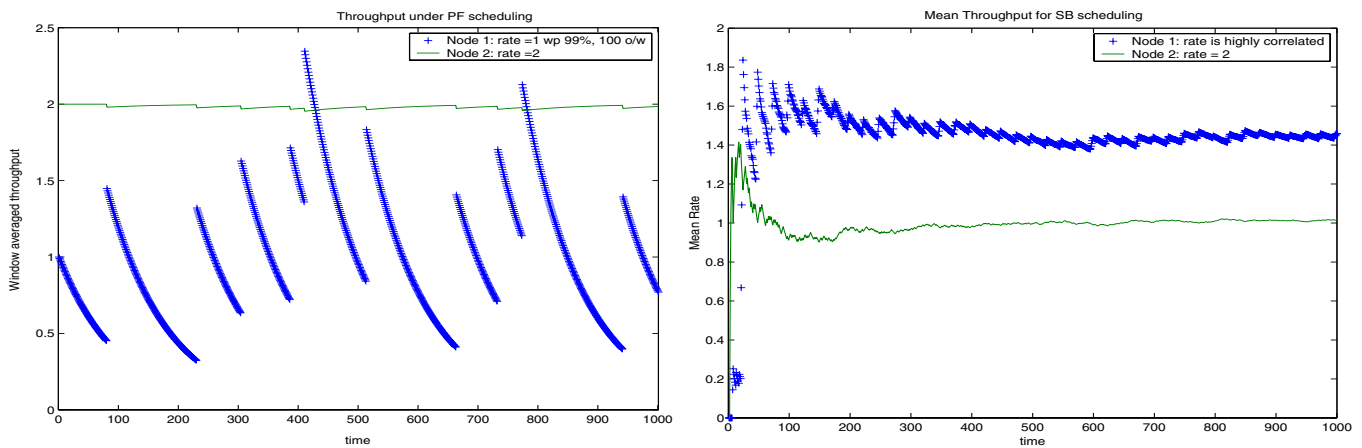


Fig. 1. PF and SB in two simple examples

vary from one node to the next as the underlying reference point is dependent on which node is being considered¹.

To illustrate this, let's look at two simple worst case examples. Take two nodes sharing one channel. One node, say 1, can achieve a rate of 1 w.p. 0.99 or 100 otherwise. The other node, node 2, always achieves a rate of 2. This is illustrated in Figure 1, which plots R_1 and R_2 according to equation 1. $\alpha = 0.01$ in the numerical values.

PF will roughly work in the following manner: when the rate of node 1 is 100, then it is chosen. Otherwise node 2 is chosen, as long as the scheduler remembers that node 1 has achieved this rate of a 100 (how long the scheduler remembers depends on the parameter of the moving average used to estimate the rate of node 1. On the figure, $\alpha = 0.01$). Of course, this prevents node 1 from transmitting most of the time. In the long run, node 2 achieve an average throughput of 2, while its mean rate is 2 as well. Node 1 on the other hand, achieves a rate close to 1, while its mean rate is close to 2 too. PF fails at providing a throughput that follows the same proportionality relationship to the mean rate requests for both nodes. It is half for node 1, and 1 for node 2.

For a given channel, what matters for the decision is whether or not this channel is performing beyond its expectations. Proportional fairness assesses the performance by dividing the current achievable rate by the mean throughput. However, comparing to the achieved throughput is not reliable in this situation, due to the heterogenous fading characteristics of the two radio channels.

¹The SB scheduler reference becomes the same for all node as W goes to infinity. This is obviously impractical.

The score-based (SB) scheduler attempts to solve this problem, but it is not immune to a different issue. Consider a different example with 2 nodes as well. Node 1 can achieve the rate sequence of 1 twenty times in a row, then 10 five times in a row. Node 2 can achieve a constant rate 2. If the depth of the scheduler is 3, then node 1 and 2 roughly get chosen with the same probability except for the first two times after node 1 transitions from one rate to the other one. The mean rate achieved in this case is depicted on the right hand side of figure 1. The correlation reduces the effect of the scheduling. We see that node 2 achieves rate 1 and that node 1 achieves rate close to 1.4. This is about the performance of a round robin system, that is, of a non-opportunistic scheduler. A scheduler that chooses node 1 when its rate is 10 and node 2 otherwise would achieve rate 1.6 for node 2, and rate 2 for node 1, both significantly improving on SB. This worst case example illustrates the sensitivity of the SB scheduler to correlation of the rates. But this is this very correlation that opportunistic schedulers attempt to leverage: it is because the time scale for the fading variations is longer than the time to make the scheduling decision, that opportunistic scheduling brings about a performance gain. Correlation of the rate requests are inherent to the system, and the issue needs to be addressed.

In the two simple example, we saw that the scheduling algorithms performed unevenly with respect to the objective they set to achieve. In both case, it is because the reference framework is modified by the scheduler with the new rate information received in each time slot. In this paper, we suggest to construct an external reference, and to base the scheduling decision depending on how the new rate

information fits within this reference.

In the next section, we describe how to construct a series of rate bins for each node, so that the relation of the rate value to these bins is the same for all nodes: this is our external reference framework. We also show how to make a scheduling decision based on these rate bins. We prove that our scheduling decision follows the same rule for all the nodes independently of the fading characteristics of the channels. In section IV we then present some numerical results to show the gain in fairness and the performance evaluation of the scheduling algorithm.

III. QUANTIZED SCHEDULING

A. Quantiles definition

We are trying to construct a reference framework that is an invariant of the rate distribution of the channel. We consider a system with N nodes. As a first step, we want to be able to define Q quantiles for each node. A quantile set for node i is a partition of \mathbf{R}^+ made of Q continuous rate sets (typically intervals) $B_{i,q}$, $q = 1, \dots, Q$ such that, for the distribution μ_i of the rate r_i ,

$$\mu_i(B_{i,q}) = 1/Q \quad (2)$$

The intervals $B_{i,q}$ are the *rate bins* for the distribution μ_i . When node i requests rate $r_i(t)$ in the time slot t , we assign it a value $q_i(t)$ based on which quantile bin $r_i(t)$ falls into. q_i is the quantile index. The scheduling decision $\sigma(t)$ is then:

$\sigma(t) = \arg \max_{i \in \{1, \dots, N\}} q_i(t)$ where

$$r_i(t) \in B_{i,q_i(t)} \quad (3)$$

with a uniform random tie breaker. So, if the value of $r_i(t)$ falls into the top $100/Q$ % for the rate distribution of node i , we assign it the value Q . If it falls into the second $100/Q$ %, then we assign it the value $Q - 1$, etc, until it falls into the lowest $100/Q$ % of the rate distribution of node i , where it is assigned the value $q_i(t) = 1$.

Remark III.1: : The value of $q_i(t)$ need not to be a linear function of the quantile index. It is possible to construct other functions. The only property that is required is that it is increasing with the quantile index.

Assuming the rate bins are given as in equation (2), the value $q_i(t)$ is a uniform random variable on $\{1, \dots, Q\}$, from which all the nice properties of

the SB scheduler can be derived, especially in terms of performance in heterogenous channels. Fairness is achieved as all the nodes compete for scheduling using a value q_i that is homogenous from the point of view of the scheduler. The fact that the value of q_i increase with the rate performance ensures the correct opportunistic use of the channel.

We now describe how to compute the quantiles for the rate distribution μ_i of r_i . The description is similar for each node, so we drop the subscript i in the next section whenever there is no confusion.

We assume that the rate distribution is bounded by a value R_{max} , as it simplifies the description of the algorithm. However, the results can be easily extended to the case $R_{max} = \infty$. We first consider a special case, namely when μ does not have any singularity.

B. Particular case

Since μ does not have any singularity, the cdf of r_i is strictly increasing. For node i , we give ourselves a sequence of rate values $\rho_j(0)$, $j = 0, \dots, Q$ such that $\rho_0(0) = 0 < \rho_1(0) < \dots < \rho_{Q-1}(0) < \rho_Q(0) = R_{max}$

Thus, the $\rho_j(0)$ form a partition of $(0, R_{max})$. For every rate $r_i(t)$ that falls into $(\rho_j, \rho_{j+1}]$, we assign value j to the rate $r_i(t)$ and increase a counter γ_j by one. After T time intervals, we recompute the value of the ρ sequence by using the *bin update* algorithm.
Bin Update Algorithm:

- Compute the values $V_j = \sum_{k=1}^j \gamma_k / T$, with $V_0 \triangleq 0$, and $j = 1, \dots, Q$. Define the function $v(x)$ to be the linear interpolation² of the points (ρ_j, V_j) for $x \in (0, R_{max})$.
- Set $\rho_j(T) = v^{-1}(j/Q)$ for $j = 1, \dots, Q - 1$, where $v^{-1}(x) \triangleq \min_y \{v(y) = x\}$.
- Reset the counters γ_j back to 0.

Figure 2 illustrates the algorithm.

Note that the state to be maintained is composed of $Q - 1$ values for the ρ_j 's and Q counters. The convergence speed of the bin update algorithm can be increased by keeping track of the previous points $(\rho_j((n-1)T), V_j((n-1)T))$ at the previous update time $(n-1)T$ so as to refine the interpolation function $v(x)$ at time nT with $2Q$ linear segments instead of Q . This avoids the convergence to ping-pong around the limit.

²more complicated interpolations than linear are possible of course. Simulations later show that linear seem to work well enough

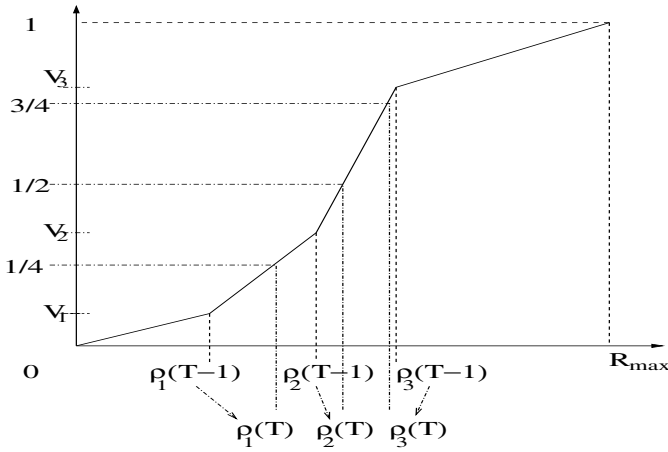


Fig. 2. Bin Update Algorithm

Because μ does not have any singularity, the cdf is a continuous increasing function on $[0,1]$ and the ρ_j 's converge to value ρ_j^* 's such that rate $r_i(t)$ falls into each bin (ρ_j^*, ρ_{j+1}^*) with equal probability. This also implies some increasing property on the quantile value q_i . q_i is increasing (however not strictly) increasing in r_i . The set B_q in equation 2 can thus be set as (ρ_{q-1}^*, ρ_q^*) .

C. General case

In general, the algorithm in the previous case will not necessarily converge to equiprobable bins if μ has singularities. In the extreme case of a point mass for rate r , ie. $\mu = \delta_r$, all the ρ_j will converge to r and only one counter γ_j accounts for the different rate requests.

Also, even if μ has no singularity, the empirical bins constructed according to the bin update algorithm might be inaccurate, depending on the value of the update time T . The shorter T , the less accurate the constructed bins. To correct from the inaccuracy due to the empirical estimations or the singularities in the distribution, we use the *quantized scheduling* algorithm:

The values V_j specify a partition of $[0, 1]$. However this partition differs from the one created by the intervals $[(j-1)/k, j/k)$. Define U_j to be the interval $[V_{j-1}, V_j)$ for $j = 1, \dots, Q$. Similarly, define I_j to be the interval $[\frac{j-1}{Q}, \frac{j}{Q})$ for $j = 1, \dots, Q$. The length of a set S is defined as: $l(S) = \max(x \in S) - \min(x \in S)$ and $l(\emptyset) \triangleq 0$. Define the probabilities π_k^j to be, for all U_k with non-null length:

$$\forall j, k \in \{1, \dots, Q\}, \pi_k^j = \frac{l(U_k \cap I_j)}{l(U_k)} \quad (4)$$

If U_k has length zero, then it means that no rate fell in this interval, so there is no need to compute the value π_k^j . We can check that $\sum_j \pi_k^j = 1$ for all j , as the U_k partition $[0, 1]$.

The rate $r_i(t)$ belongs to some set $B_k \triangleq [\rho_{k-1}, \rho_k)$. We define the value $q_i(t)$ to be equal to j with probability π_k^j . The quantized scheduling algorithm thus acts as follows (illustrated in figure 3).

Quantized Scheduling Algorithm:

- at every sampling update nT , compute the value of the π_k^j based on the values of V_i 's and γ_i 's.
- for each rate $r_i(t)$, find the interval B_k it belongs to, and the corresponding index k .
- Generate a random variable equals to j with probability π_k^j . $q_i(t) = j$
- Select the node with the maximal q_i . Break the ties with an equiprobable coin toss.

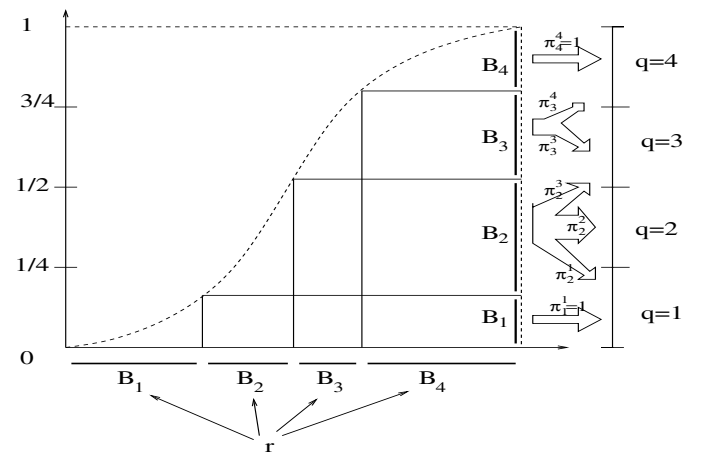


Fig. 3. Quantized Scheduling Algorithm

The algorithm is using the bin update algorithm as in the particular case for the computation of the values ρ_j . For determining the value q_i , the computation of the probability π_k^j is added. Note that if I_k is close to U_k , then $\pi_k^k \simeq 1$ and the general case is an extension of the particular case: whenever the rate $r_i(t)$ falls into the bin $(\rho_{k-1}, \rho_k]$, it receives the value $q_i(t) = k$.

Theorem III.1: $q_i(t)$ is uniformly and independently distributed over $\{1, 2, \dots, Q\}$ when $T \rightarrow \infty$

Proof:

$$\begin{aligned} P(q_i(t) = j) &= \sum_k P(r \in B_k, q = j) \\ &= \sum_k \pi_k^j \mu(B_k) \end{aligned} \quad (5)$$

$$l(U_k) \rightarrow_{T \rightarrow \infty} \mu(B_k) \text{ as } V_j \rightarrow P(r < j/Q) \text{ for } T$$

large enough. Using 4 yields

$$P(q_i(t) = j) = \sum_k l(U_k \cap I_j) = \frac{1}{Q} \quad (6)$$

where the last equality stands from the fact that the U_k partition $[0,1]$ and $l(I_j) = 1/Q$. ■

This scheduling algorithm produces for node i a random value that is uniformly distributed over $1, \dots, Q$, and is tightly correlated to the performance of the channel: the value is quasi-increasing with the channel conditions. It is not strictly increasing: if $r < r'$ are such that they both belong to $(\rho_{j-1}, \rho_j]$, then the values q could be higher than q' . However, if $r < r'$ fall in different bins, then $q \leq q'$.

Remark III.2: We chose Q bins and Q possible outcome for the decision variable $q_i(t)$. This in turn implied a square (π_k^j) matrix. The particular case implies a diagonal matrices where $\pi_k^k = 1$ and $\pi_k^j = 0$ for $j \neq k$. We could have considered S values for the $\rho_j, j = 1, \dots, S$ with $\rho_0 = 0$. This would yield S interval I_k , and a $S \times Q$ matrix for the π_k^j .

Remark III.3: If $R_{max} = \infty$, the algorithm can be adapted as follows. Choose $\rho_Q(0)$ arbitrarily, yet such that $\rho_Q(0) > \rho_{Q-1}(0)$. At every update time nT , find k such that $k = \max\{j : j/Q \leq v(\rho_Q)\}$. If $k = Q$, then update as in the bin update algorithm. If $k < Q$, then update ρ_1, \dots, ρ_k as in the quantile update algorithm. For $j = k+1, \dots, Q$, set $\rho_j = \alpha^{j-k} \rho_k$ with $\alpha > 1$. Note that it does not matter whether or not ρ_Q eventually converges to ∞ as we are only interested in finding ρ_{Q-1} such that $P(r_i > \rho_{Q-1}) = 1/Q$. Once we find a value ρ_Q such that $v(\rho_Q) > (Q-1)/Q$, then we can converge to ρ_{Q-1} .

Remark III.4: The quantized scheduler can of course be used without the bin update algorithm. The rate bins B_j can be preset once and for all as a partition of $[0, \infty)$. The values γ_j and V_j can then be replaced by exponential moving averages of the form:

$$V_j = \beta V_j + (1 - \beta) \mathbf{1}_{\{r_i \in \cup_{i=1}^j B_i\}}$$

which converges to $P(r_i \in \cup_{i=1}^j B_i)$. The intervals U_k 's and probabilities π_k^j 's are derived identically as in the previous section. However, since the sets B_j 's are not updated, the initial condition will strongly influence the eventual performance of the system. For instance, if all rates r_i always fall in the same single bin B_z , then q_i becomes an i.i.d. r.v. on $\{1, \dots, Q\}$ that is no longer correlated with r_i : the system becomes equivalent to a round-robin system.

D. Slotted rates

The general case above can be transposed for slotted rates. Assume now that the rate $r_i(t)$ can take only a finite number M of values. This is the case for most cellular communication systems. The quantized scheduling algorithm can be applied here. Define the possible rates to be R_1, \dots, R_M .

Define $\rho_0 = 0$, and then $\rho_j \in [R_j, R_{j-1})$. By the remark III.2, we can construct the probabilities π_k^j as in the previous case, for $k = 1, \dots, M$ and $j = 1, \dots, Q$. There is no need to update the value of ρ_j , as the description of the rate distribution μ cannot be improved upon.

In the slotted case, if r and r' fall in the same bin, they share the same discrete value R_j . Thus the value $q_i(t)$ is increasing in $r_i(t)$.

In practice, the value ρ_j would not be computed. The counters γ_j would be associated with the rate R_j and the quantized scheduling algorithm would use the values of γ_j 's to compute V_j 's, B_j 's and π_k^j 's.

E. QoS

We have shown how to construct a uniform random variable that is quasi-increasing with $r_i(t)$. However, the choice of a uniform distribution for the value of q was dictated by fairness among the nodes competing for the channel. It is possible to modulate the distribution in order to enforce some level of QoS.

For instance, a possible QoS policy would give uniform values for best effort nodes. Nodes that request a better level of service could receive a distribution slanted towards higher values. Instead of having Q quantiles of equal size, the nodes receiving preferred service would have quantiles such that the top ones are larger than the bottom one. There are infinitely many ways of doing so.

IV. NUMERICAL RESULTS

We simulate the algorithm here and compare it with respect to the proportional fairness (PF) algorithm. We assume that $R_{max} = \infty$ for the simulations, whenever we use a Rayleigh distribution. Since we are searching for the rate that define the different quantiles, the value of the lower and higher rates do not matter, as long as most of the distribution density falls within them. We consider the nodes to have different Rayleigh parameters σ . We take

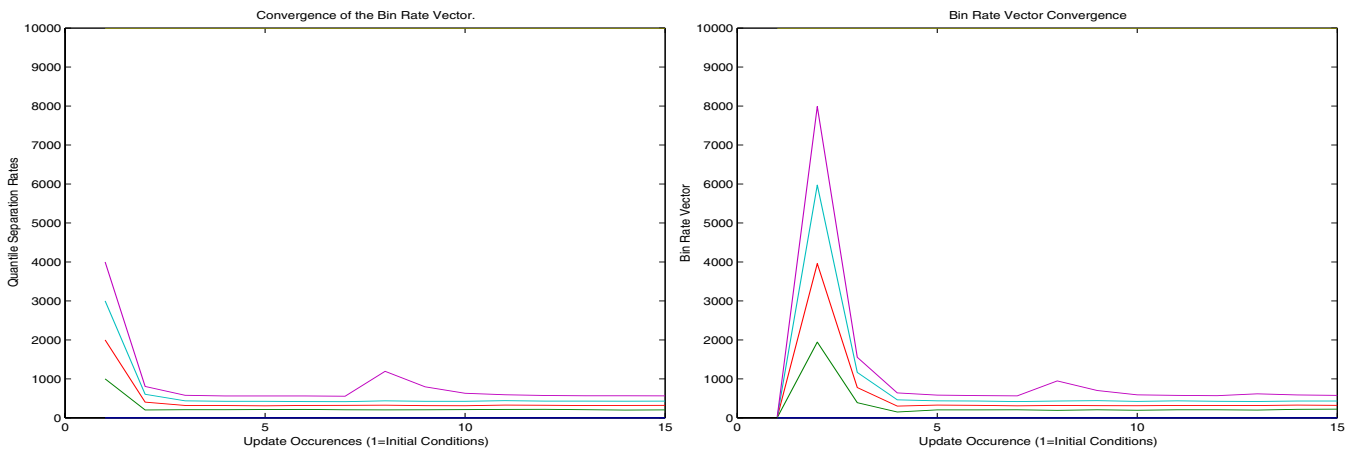


Fig. 4. Convergence of the Bin Rates for different sets of initial conditions

the same representative values for the values of σ as in [7].

We first simulate the bin update algorithm. Figure 4 describes the rates ρ_i 's for one node. Since all nodes follow a Rayleigh channel rate distribution, the behavior is symmetrical along all nodes. The initial rate vector on the left hand side is $[0, 1000, 2000, 3000, 4000, 10000]$. For a node with $\sigma = 316$ and 5 quantiles, the last rate vector is $[0, 201.2, 318.8, 427.3, 566.7, 10000]$ and the theoretical rate vector is:

$$\rho_i = \sqrt{-2\sigma^2 \ln(1 - i/N)} \quad (7)$$

that is, $[0, 211.7, 320.3, 429.0, 568.5, \infty]$. We varied the initial rate vectors on the right hand side to $[0, 10, 20, 30, 40, 10000]$. The last bin rate vector is $[0, 223.9, 321.6, 431.5, 575.8, 10000]$. Again, this converges to the theoretical value.

Figure 5 shows the comparison of proportional fairness with respect to the Quantized Scheduling (QS) in Rayleigh channel. We simulate the QS algorithm with a number of quantiles varying between 7 and 20. We can see that PF slightly outperforms QS, but that for increasing number of quantiles, the difference disappears. The reason is that the quantized scheduling lumps into the same top quantile rates that vary from ρ_{Q-1} to ∞ , such that two nodes that for instance achieve respectively $\rho_{Q-1} + \epsilon$ and $10^6 \rho_{Q-1}$ are chosen with the same probability by the QS algorithm. PF on the other hand will select the second one, and achieve a better performance. As the number of quantiles increases, the likelihood that both nodes request a rate in the top quantile decreases.

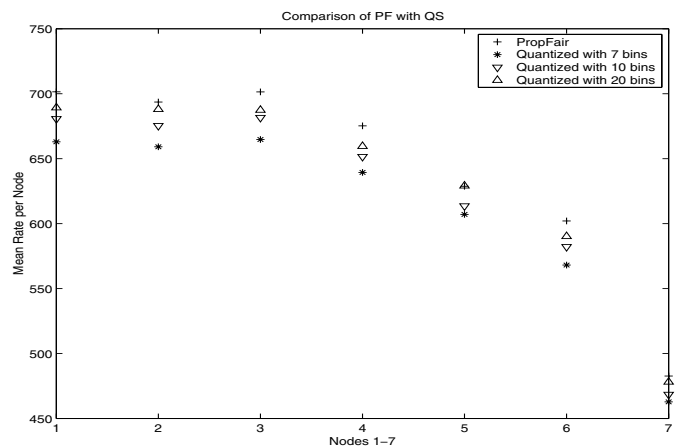


Fig. 5. Proportional Fairness vs. Quantized Scheduling in Rayleigh Channels

On the other hand, in practical networks, rates cannot increase to be infinitely large. In such case, QS performs as well as PF in homogeneous environments, and better than PF in environment where different channels have different rates distribution. Figure 6 shows the performance of QS compared to PF in a slotted rate environment. The performance here is strictly similar.

In the heterogenous case, the gain shown in the simulations of [6] applies exactly, as the QS is strictly equivalent to the score-based approach in this case: the value of the decision variable is a uniform random variable on $[0, Q]$ here.

We conducted some simulations to compare Quantized Scheduling with Proportional Fairness. We assumed nodes of two types were connected to the base station: one node with a Rayleigh channel and the rest of the nodes with a perfect constant channel. We varied the total number of nodes.

In Figure 7 we show the normalized throughput

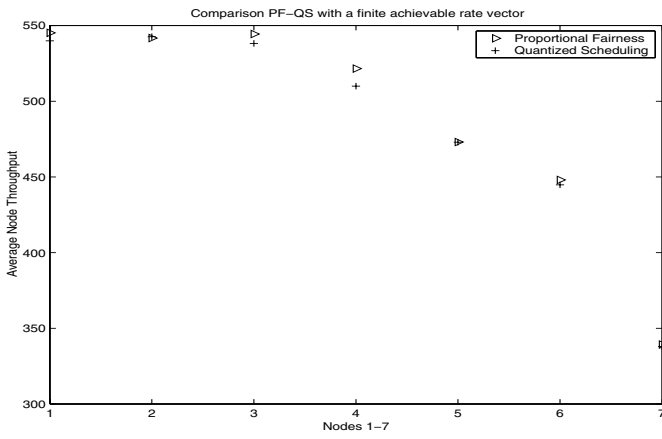


Fig. 6. Proportional Fairness vs. Quantized Scheduling in Rayleigh Channels with Slotted Rates

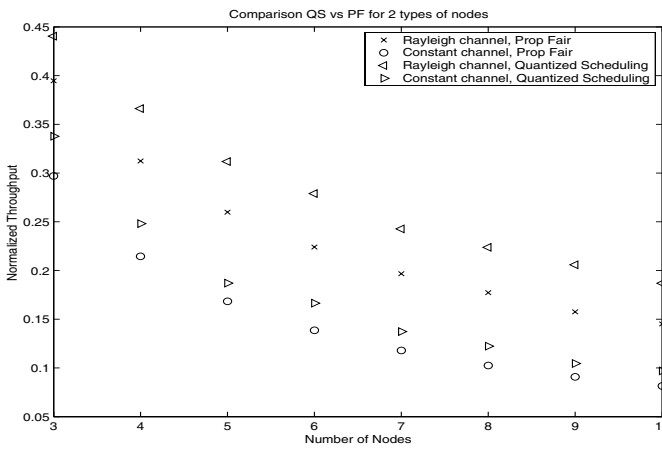


Fig. 7. Proportional Fairness vs. Quantized Scheduling in Heterogeneous Channels with Slotted Rates

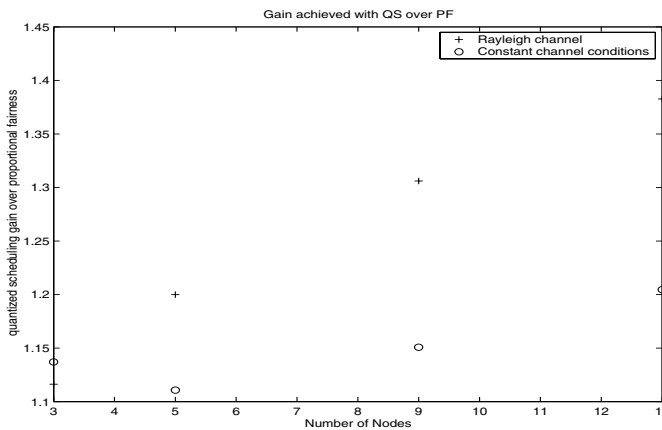


Fig. 8. Gain of Quantized Scheduling over Proportional Fairness in Heterogeneous Channels

(that is the achieved rate divided by the mean channel rate) for the two types of nodes. We see that QS outperforms PF for both types of nodes. Figure 8 illustrates this by plotting the ratio of the throughput

for a QS node of each type, divided by the corresponding throughput achieved with PF.

The impact on fairness can be computed in many ways. QS is fair by its definition, as each node as the same likelihood of being chosen: the decision variable for each node are i.i.d. random variables over the same state space. This means that each node will be chosen the same number of times. However, the achieved throughput might not be fair in heterogeneous channels. We define the throughput fairness index to be:

$$F(\Theta) = \frac{(\sum_{i=1}^N \Theta_i)^2}{N \sum_i \Theta_i^2} \quad (8)$$

where Θ_i represents the variable whose fairness we are trying to assess. A value of 1 for F means perfect fairness, while a value of $1/N$ means perfect imbalance. Figure 9 plots the throughput fairness index (where Θ is the achieved throughput for each node). As we can see, even though the throughput for QS is better by 10% to almost 40% in Figure 8, the throughput fairness is within 3% for both compared scheduling algorithms.

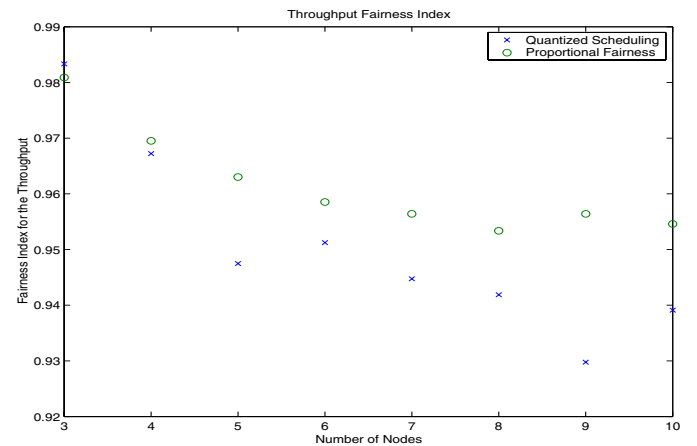


Fig. 9. Throughput Fairness Index PF vs. QS in Heterogeneous Channels

V. CONCLUSION AND FUTURE WORK

The contribution of this paper is to introduce a quantized scheduling algorithm that constructs a partition for the channel distribution, and uses this partition to estimate whether the channel is under- or over-performing. We also have shown that this scheduling can be used when the partition is given by the system itself, as in a rate slotted case.

We compared the algorithm with PF algorithm and SB algorithm and showed that the quantized scheduling algorithm does not suffer from the distribution bias in heterogenous systems as does PF scheduling, nor from the incidence of rate correlation as with SB scheduling.

The next natural step would be to apply the framework in [8] to derive theoretic gains for this scheduling algorithm, and see how close to the maximal theoretical gain it is. However, as it is quasi-equivalent to PF in homogenous case, and better in the heterogenous case, the SQ policy must achieve a gain close to the maximal gain.

REFERENCES

- [1] P. Bender, P. Black, M. Brob, R. Padovani, N. Sindhushayana, A. Viterbi, *CDMA/HDR: a Bandwidth-Efficient High-Speed Wireless Data Service for Nomadic Users*, IEEE Communications Magazine, July 2000, pp.70-7.
- [2] E. Esteves, *The High Data Rate Evolution of the CDMA2000 Cellular System*, Multiaccess, Mobility and Teletraffic for Wireless Communications. Vol. 5, pp.61-72, Kluwer Academic Publishers, 2000.
- [3] A. Bedekar, S.C. Borst, K. Ramanan, P.A. Whiting, E.M. Yeh, *Downlink scheduling in CDMA data networks*, Proc. IEEE GLOBECOM'99, or extended version: Technical Report PNA-R9910, CWI.
- [4] 3rd Generation Partnership Project 2 "3GPP2", *3GPP2 Specifications*, http://3gpp2.org/Public_html/specs/index.cfm.
- [5] J.M. Holtzman, *Asymptotic analysis of proportional fair algorithm*, Personal, Indoor and Mobile Radio Communications, 2001 12th IEEE International Symposium on , 30 Sept.-3 Oct. 2001, pp. F-33 -F-37 vol.2.
- [6] T. Bonald, *A Score-based Opportunistic Scheduler for Fading Radio Channels*, Proceeding of the European Wireless Conference, Barcelona, March 2004.
- [7] S. Shakkotai, 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, Vol. 207, 2002.
- [8] S. Borst, *User-level Performance of Channel-Aware Scheduling Algorithms in Wireless Data Networks*, in Proceedings of IEEE INFOCOM 2003, San Francisco.