

**Channel Predictive Proportional Fair Scheduling**

Journal:	<i>IEEE Transactions on Wireless Communications</i>
Manuscript ID:	Paper-TW-Sep-06-0729
Manuscript Type:	Original Transactions Paper
Date Submitted by the Author:	20-Sep-2006
Complete List of Authors:	Bang, Hans Jørgen; UniK Ekman, Torbjörn; NTNU Gesbert, David; EURECOM
Keyword:	Resource management and QoS provisioning < Multimedia, Networks and Systems, Cross-layer design and optimization < Multimedia, Networks and Systems

Channel Predictive Proportional Fair Scheduling

Hans Jørgen Bang, Torbjörn Ekman and David Gesbert

Abstract

Recent work on channel modeling and prediction has shown the feasibility of predicting the mobile radio channel, quite accurately, several milliseconds ahead in time for realistic Doppler spreads. Motivated by these results we consider opportunistic scheduling algorithms that exploit both current and future channel estimates. We demonstrate how this extra channel information can be used to improve the scheduling. Simulations show that the proposed algorithm can improve the inherent tradeoff between throughput, fairness and delay. The current approach builds on proportional fair scheduling but can also be generalized to other criteria.

Index Terms

Multiuser diversity, proportional fair scheduling, fairness, latency, channel prediction.

I. INTRODUCTION

Opportunistic scheduling has recently attracted much attention as a means to increase the spectrum efficiency of wireless data networks. By allocating the common radio resource to the users that are currently best able to utilize it the inherent multiuser selection diversity is exploited [1], [2]. The multiuser diversity results from the fact that different users usually have

Corresponding author: H. Bang is with UniK - University Graduate Center, University of Oslo, Instituttveien 25, P. O. Box 70, N-2027 Kjeller, Norway, email: hans@unik.no.

T. Ekman is with the Department of Electronics and Telecommunications, Norwegian University of Science and Technology, 7491 Trondheim, Norway, email: torbjorn.ekman@iet.ntnu.no.

D. Gesbert is with the Mobile Communications Department, Eurcom Institute, 2229 Route des Crtes, BP 193, F-06904 Sophia Antipolis Cdex, France, email: david.gesbert@eurecom.fr.

This work was presented in part at SPAWC'2005, New York, June 2005.

independent channel gains. Thus, in a system with several fading users the probability a user with a good channel is always high. There is however a fundamental tradeoff between throughput on one hand and fairness and delay on the other. Techniques that are able to push this tradeoff (improving one without sacrificing the other) are therefore of great interest.

Several channel-aware scheduling techniques for the downlink of wireless data networks have already been proposed in the literature. One of the most popular ones is the *Proportional Fair Scheduler* (PFS) [2]. In this approach a time constant parameter is chosen to specify over what time period fairness between the users should be maintained.

In parallel and much independently, recent work on channel modeling and prediction has shown the feasibility of predicting fading channels over horizons of up to 0.25 wavelengths with reasonable accuracy [3]–[5]. This opens up the possibility of combining channel prediction with resource allocation. In this paper we examine how channel predictions can help to improve the tradeoff between throughput and fairness. To the best of our knowledge, this idea has not been investigated before. More specifically we develop a framework for scheduling algorithms that rely on both current and future channel estimates. The intuition behind our approach is that future channel estimates, even imperfect ones, are beneficial since they allow the scheduling to be planned over several time slots ahead in time. We make the following key points:

- For scenarios where fairness is to be maintained over long periods of time compared to the coherence time of the fading, capacity maximization can be obtained through the use of memoryless schedulers. Thus, channel prediction is of minor interest¹.
- For scenarios where tighter fairness and delay constraints are used, there is a significant gain in terms of throughput to be obtained from a channel prediction-aware scheduler.

Clearly, there are some major challenges associated with prediction-aware scheduling. First, the quality of the predictions degrade rapidly with the prediction horizon. Thus, robustness with respect to prediction errors is crucial. Second, the complexity of the scheduling tends to increase significantly when additional channel information is introduced. In this paper we address both these issues. Specifically, we propose a generalization of the PFS algorithm that incorporates future channel estimates. We demonstrate that this algorithm is capable of increasing

¹In a practical system channel prediction must still be employed to obtain estimates of the users' current channel conditions due to non-zero feedback delay

the throughput without compromising fairness and delay compared to the standard PFS algorithm.

The rest of this paper is organized as follows. In Section II we introduce the system model and review channel-aware scheduling. In Section III we discuss prediction-aware scheduling. Specifically we extend the PFS algorithm to a prediction-aware scheduling scenario. In Section IV we evaluate the proposed algorithm in terms of throughput, fairness and delay through simulations. Finally we conclude the paper in Section V.

II. SYSTEM MODEL AND CHANNEL-AWARE SCHEDULING

A. System Model

We consider the downlink of a single cell with N simultaneously active users served by one base station (BS). The scheduling is organized on a slot by slot basis, i.e. one and only one user is served during any given slot. The scheduler resides at the BS and decides prior to each slot which user the BS shall transmit data to. We use $i^*(k)$ to denote the user scheduled in slot k .

The BS operates at fixed transmit power and employ rate adaption to adjust to instantaneous channel conditions. Our key assumption is that estimates of the users' supportable data rates for the current and $L - 1$ future slots are available to the scheduler. The supportable rate for the i th user in slot $k + l$, as predicted in slot k , will be denoted $\hat{R}_i(k + l|k)$. We use $R_i(k)$ as shorthand for $\hat{R}_i(k|k)$. For an arbitrary vector \mathbf{i} we let $(\mathbf{i})_l$ denote its l th component.

B. Channel-Aware Scheduling

In order to harness the multiuser diversity the scheduling criterion must be a function of the users' current supportable rates. An obvious example of a channel-aware scheduler is the Max SNR scheduler which always selects the user with the highest supportable transmission rate, i.e.

$$i^*(k) = \arg \max_{i=1,..,N} R_i(k). \quad (1)$$

The Max SNR scheduler is important since it maximizes the users' combined data rate and therefore gives an upper bound on the realizable multiuser diversity gain. Note that the Max SNR scheduler is memoryless in the sense that current scheduling decisions are independent of past scheduling decisions. In general channel-aware memoryless schedulers can be written in the form

$$i^*(k) = \arg \max_{i=1,..,N} f_i(R_i(k)), \quad (2)$$

where $f_i(\cdot)$ is a function independent of time k . The main rationale for not scheduling the user with the highest supportable rate directly as in (1) is to ensure a fair resource allocation between the users even if their mean SNR differ. Recently several memoryless channel-aware scheduling policies that maximizes capacity under various long-term fairness criteria have been proposed [6]–[8].

C. Proportional Fair Scheduling

A main limitation with memoryless schedulers is that fairness can only be ensured over long time windows compared to the coherence time of the fading. In order to control delay and ensure fairness over smaller time horizons memory has to be introduced in the scheduler. By introducing memory the priority of users that has not been served for a long time can be raised. This is exemplified by the PFS scheduler. In this approach the user with the highest supportable rate relative to its past average throughput is selected in each time slot. Thus, the user scheduled in time slot k is given by

$$i^*(k) = \arg \max_{i=1,\dots,N} \frac{R_i(k)}{T_i(k)}, \quad (3)$$

where $T_i(k)$ is user i 's past average throughput. The average throughputs are updated in each time slot according to

$$T_i(k+1) = \left(1 - \frac{1}{t_c}\right)T_i(k) + \frac{1}{t_c}R_i(k)\delta(i - i^*(k)), \quad (4)$$

where $\delta(\cdot)$ is the Kronecker delta function and t_c is a pre-determined constant. The particular value of t_c determines the time horizon over which the throughputs are computed and gives a tradeoff between long-term throughput and delay. The long-term throughput of user i is defined as

$$\liminf_{K \rightarrow \infty} \frac{1}{K} \sum_{k=1}^K R_i(k)\delta(i^*(k) - i), \quad (5)$$

and differs from the running averages defined in (4).

Even though the above scheduling criterion can be easily motivated there exist a second formulation that provides additional insight into the nature of the PFS algorithm [9]–[11]. To this end we consider the following system utility function

$$\mathcal{U}(k) = \sum_{j=1}^N \log T_j(k). \quad (6)$$

It can then be shown (see *Proposition 2*) that (3) is equivalent to selecting the user that leads to the largest instantaneous increase in \mathcal{U} . More precisely this can be formulated as

$$i^*(k) = \arg \max_{i=1,\dots,N} \mathcal{U}(k+1). \quad (7)$$

This suggests that the underlying goal of the PFS algorithm is to maximize the system utility function \mathcal{U} .

III. PREDICTION-AWARE SCHEDULING

In order to further improve the performance of channel-aware schedulers, we consider prediction-aware schedulers. This class of schedulers exploit both current rate estimates and predictions of the users' future supportable rates.

To illustrate how channel predictions can be utilized we first consider the following simple example. Assume two users with supportable rates in time slot k and $k+1$ as illustrated in Fig. 1. Suppose now that we want to maximize the sum throughput while maintaining fairness over the same time interval, we can then easily argue that user 1 should be allocated the first time slot and user 2 the second slot. However, to ensure that the right scheduling decision is made in time slot k estimates of the supportable rates in time slot $k+1$ must be available.

Prediction-aware schedulers can be loosely classified according to if the scheduling decisions are made simultaneously for a block of time slots or if the decisions are made one at a time. We first consider a block-based strategy. In this approach adjacent time slots are grouped into non-overlapping blocks of length L time slots. Every L th time slot (say time slot k) a scheduling vector

$$\mathbf{i}^*(k) = \{i^*(k+l)\}_{l=0}^{L-1} \quad (8)$$

dictating the next L transmissions is computed according to

$$\mathbf{i}^*(k) = \arg \max_{\mathbf{i} \in \mathcal{F}} \mathcal{O}^{(k)}(\mathbf{i}), \quad (9)$$

where $\mathcal{O}^{(k)}(\mathbf{i})$ is an appropriate objective function and \mathcal{F} is the set of feasible scheduling combinations. In time slot $k+L$ a new scheduling vector determining the schedule for the next block of time slots is computed. The main motivation for the above approach is that the scheduling can be better planned if several scheduling decisions are made simultaneously. The objective function should reflect the underlying goals of the scheduling and can be a function of

a number of parameters including but not necessarily limited to predictions of the users future rates.

A. Predictive Proportional Fair Scheduling

In this subsection we extend the PFS algorithm to a prediction-aware scheduling scenario. We first define the following two quantities

$$\tilde{T}_i(k|\mathbf{i}) = \left(1 - \frac{1}{t_c}\right)T_i(k) + \frac{1}{t_c} \sum_{l=0}^{L-1} w_l \hat{R}_i(k+l|k) \delta(i - (\mathbf{i})_{l+1}) \quad (10)$$

$$\mathcal{O}_{pf}^{(k)}(\mathbf{i}) = \sum_{i=1}^N \log \tilde{T}_i(k|\mathbf{i}), \quad (11)$$

where w_0, \dots, w_{L-1} are positive weights in (10). With $\mathcal{O}_{pf}^{(k)}(\mathbf{i})$ as objective function we can directly adopt the block-based scheduling strategy outlined above. Note that for $w_0 = 1$ and $L = 1$ we obtain (7) and hence the PFS algorithm as a special case. For $L > 1$ we obtain a family of prediction-aware schedulers depending on the particular combination of weights w_l . The next result further illustrates the connection with the PFS algorithm.

Proposition 1: Assume that the rate predictions are perfect and let

$$\mathbf{i}^*(k) = \arg \max_{\mathbf{i} \in \mathcal{F}} \mathcal{O}_{pf}^{(k)}(\mathbf{i}), \quad (12)$$

with $w_l = \left(1 - \frac{1}{t_c}\right)^{-l}$ for $l = 0, \dots, L - 1$. Then $\mathbf{i}^*(k)$ maximizes $\mathcal{U}(k + L)$.

Proof: See Appendix 1. ■

Thus, for the weights w_l specified in *Proposition 1* the scheduling vector $\mathbf{i}^*(k)$ is chosen to maximize $\mathcal{U}(k + L)$. Observe that this generalizes the standard PFS algorithm where $i^*(k)$ maximizes $\mathcal{U}(k + 1)$. Even though this seems to justify this particular choice of weights one important remark must be made. Since $\left(1 - \frac{1}{t_c}\right)^{-1} > 1$ we can infer from (10) that increasing emphasis is put on more distant rate predictions. This is unfortunate given that the quality of the rate predictions typically decrease with increasing prediction horizon. From a robustness point of view the weights w_l should preferably be monotonically decreasing.

B. Robust Prediction-Aware Scheduling

A major challenge with the block-based scheduling strategy is that we rely on estimates of the future feasible rates to compute $\mathcal{O}^{(k)}(\mathbf{i})$. Reliable short range predictors have been proposed,

but the predictions degrade rapidly for longer prediction ranges [3]–[5]. To fix the schedule for a long block might therefore result in sensitivity to prediction errors. A more robust approach is to redo the schedule in each time slot and only use the first component of the scheduling vector. This results in the robust prediction-aware scheduling strategy described in Table I. Note that $\mathbf{i}^*(k)$ now represents a tentative schedule for time slot k to $k + L - 1$. Although only the first component of $\mathbf{i}^*(k)$ is explicitly used the other components will also affect the outcome of $i^*(k)$.

C. A Suboptimal Algorithm for Obtaining $\mathbf{i}^*(k)$

Another drawback of the algorithm is the complexity of the full search for the scheduling vector. In practice one may have to settle for suboptimal solutions. We next present a low complexity algorithm to be used together with the robust prediction-aware scheduling strategy that renders good but possibly suboptimal scheduling vectors $\mathbf{i}^*(k)$. We note that $\mathbf{i}^*(k)$ will still denote the computed scheduling vector even though it might not be optimal according to (9).

As opposed to an exhaustive search for the optimal scheduling vector we propose a cyclic coordinate ascent-type algorithm. At each iteration one component of the scheduling vector is updated with the other components held fixed. To describe the algorithm we use the following notation:

- $\mathbf{i}^n(k) = (i_1^n(k), \dots, i_L^n(k))$ denotes the computed scheduling vector after n iterations.
- For an arbitrary vector $\mathbf{i} = (i_1, \dots, i_L)$, $\mathbf{i} \leftarrow^l i$ denotes the vector \mathbf{i} with the l th component exchanged with i . Thus $\mathbf{i} \leftarrow^l i = (i_1, \dots, i_{l-1}, i, i_{l+1}, \dots, i_L)$.

The resulting algorithm is given in Table II and further illustrated in Fig. 2. Observe that, for each iteration, we either obtain the same or a better solution in the sense

$$\mathcal{O}^{(k)}(\mathbf{i}^{n+1}(k)) > \mathcal{O}^{(k)}(\mathbf{i}^n(k)). \quad (13)$$

Hence the algorithm will necessarily converge to a maximum since there are only a finite number of scheduling combinations. Observe further that we use the schedule computed in the previous time step to initialize the search². We can therefore expect fast convergence as limited amount of new channel state information is introduced in each time step.

²The last component of $\mathbf{i}^0(k)$ is set to 1. This particular value is arbitrary and will not affect subsequent iterations.

We next present a useful result that shows that the computational complexity can be further reduced with (11) as objective function.

Proposition 2: Let $\mathcal{O}_{pf}^{(k)}(\mathbf{i})$ be used as objective function in (21). Then (21) is equivalent to

$$i_l^{n+1}(k) = \arg \max_{i=1,\dots,N} \frac{\hat{R}_i(k+l-1|k)}{\tilde{T}_i(k+L|\mathbf{i}^n(k) \stackrel{l}{\leftarrow} 0)}. \quad (14)$$

Proof: See Appendix. ■

Thus, according to *Proposition 2*, it is sufficient to compute the ratios in (14) in each iteration. This results in considerable computational savings compared to explicitly evaluating the function $\mathcal{O}_{pf}^{(k)}(\mathbf{i}^n(k) \stackrel{l}{\leftarrow} i)$ as suggested in (21).

IV. NUMERICAL RESULTS AND DISCUSSION

In this section, we evaluate the robust version of the predictive PFS algorithm together with the iterative algorithm in Table II numerically. We consider this version of the algorithm to be of most interest because of its reduced computational complexity. For simplicity we let $w_l = 1$ for $l = 0, \dots, L-1$. The optimum weights are likely to be dependent on the quality of the predictions. The simulation results are obtained for Rayleigh fading channels with time correlations given by Jake's model. The average SNR is 0 dB and the time slot Doppler frequency product is 0.01 for each user. This means that the terminals move one wavelength in 100 time slots, e.g. 5.5 km/h with a time slot length of 1 ms in the 2 GHz band. We use the Shannon Capacity to estimate the supported data rates and we assume that the BS always have data to send to each user.

In order to generate realistic estimates of the users' future supported rates we use a linear FIR MMSE predictor with 128 coefficients to predict the future complex fading gains from past noisy observations of the channel. The channel power gain used in the Shannon Capacity is obtained as the absolute square of the predicted complex fading gain. The quality of the prediction will depend on the Channel to Estimation Error Ratio which is set to 20 dB. The NMSE for the complex fading gain prediction ten time slots ahead is roughly 10^{-2} but for one step ahead it is only 10^{-3} . Thus, the error in the estimated rate for one time slot ahead can be disregarded.

A. Qualitative Example of Predictive Scheduling

We first illustrate the performance of the predictive PFS algorithm with a qualitative example. Consider a scenario with 15 users, $t_c = 100$ and $L = 21$. Fig. 3(a) shows a snapshot of the

supported rates for one user, where a superimposed cross corresponds to an allocated slot. Note that all allocated slots are tightly clustered around local fading peaks. By comparing with Fig. 3(b), which shows the same scenario with the standard PFS algorithm, we can conclude that there appears to be significant increase in performance by using prediction. In the next subsections we validate this claim more rigorously.

B. Throughput

We next compare the total long-term throughput as a function of the parameter t_c for the standard and the predictive PFS algorithm with 15 users. The prediction horizon $L - 1$ is set to 10 slots for the predictive algorithm. On average 20 iterations were required to compute $\mathbf{i}^*(k)$ in each time slot. It can be seen from fig. 4 that there is an increase in throughput with the predictive algorithm for all values of t_c . Note however, that the largest gains (20%) occur for smaller t_c values. This is intuitive because for large values of t_c both algorithms approach the max SNR scheduler.

C. Fairness

To quantify the degree of fairness we use Jain's fairness index [12], which is defined as

$$J = \frac{(\sum_{i=1}^N x_i)^2}{N \sum_{i=1}^N x_i^2}$$

where x_i is the resource of interest allocated to user i . Jain's fairness index measures the spread in the allocated resources and will always be in the range $1/N$ to 1. It is easily verified that $J = 1$ indicates absolute fairness and $J = 1/N$ indicates no fairness, i.e all resources are allocated to one single user.

Since we assume all users to have statistical identical channels we can expect fairness between the users over longer time horizons. To measure the fairness over shorter time intervals we let $x_i = \mathcal{T}_i(w)$, where $\mathcal{T}_i(w)$ is the throughput of user i over a window of w time slots. Finally we average over all time windows of size w time slots to obtain the average Jain's fairness index. Note that by adjusting the window size w the time horizon that the fairness is computed over is adjusted accordingly.

Fig. 5 shows the average Jain's fairness index as a function of window size for the standard and predictive PFS algorithm with $t_c = 100, 1000$. Observe that the level of fairness is virtually

the same for the two algorithms. There is a negligible reduction in fairness for $t_c = 100$ and a slight increase in fairness for $t_c = 1000$ with the predictive algorithm.

Fig. 5 also shows the corresponding plots for the Round Robin and Max SNR algorithm. Not unexpectedly there is a large penalty in fairness with the Max SNR algorithm. One should also note that even though the Round Robin algorithm is perfectly fair when allocating the radio resource in time, it is not perfectly fair when considering the actual throughputs over a finite window.

D. Delay

To quantify the delay properties of the proposed algorithm we consider the lowest throughput that a single user can expect with a certain probability of outage over a finite time window. More precisely we define the *lowest achieved throughput* (LAT) for user i over a window of size w time slots with violation probability ϵ according to

$$\Pr\{\mathcal{T}_i(w) < \text{LAT}_i(\epsilon)\} < \epsilon. \quad (15)$$

As w approaches infinity $\text{LAT}_i(\epsilon)$ approaches the long-term throughput of user i , but for smaller time windows $\text{LAT}_i(\epsilon)$ will be considerably less. $\text{LAT}_i(\epsilon)$ is an interesting measure since it gives a good indication of the continuity of the incoming data flow and is tightly related to the maximum delay between two received packets for a user. Since we assume all users to have statistical identical channel we will drop the sub-index i .

Fig. 6(a) shows $\text{LAT}(0.01)$ as a function of window size for the Round-Robin algorithm, the standard PFS algorithm and the predictive PFS algorithm with $t_c = 100, 1000$. The Max SNR algorithm is not included as $\text{LAT}(0.01)$ is zero for all window sizes in the considered range. Observe that the predictive PFS algorithm performs significantly better than standard PFS algorithm, for all but very small window sizes, for $t_c = 100$. This also hold true for $t_c = 1000$ although the improvement with the predictive algorithm is more modest. Interestingly the Round Robin algorithm is outperformed by the both the standard and the predictive PFS algorithm for window sizes larger than 200 time slots.

Fig. 6(b) shows the corresponding plots for $\text{LAT}(0.1)$ with similar patterns.

V. CONCLUSION

We have extended the PFS algorithm to a scenario where estimates of the users' future rates are available. At a reasonable increase in complexity and without compromising fairness or delay the total throughput was significantly increased compared to the standard PFS algorithm. The largest gains occurred when tighter fairness and delay requirements were imposed.

APPENDIX

A. Proof of Proposition 1

Let $w_l = (1 - \frac{1}{t_c})^{-l}$ for $l = 0, \dots, L-1$. We first note that

$$\begin{aligned} (1 - \frac{1}{t_c})^{L-1} \tilde{T}_i(k|\mathbf{i}) &= (1 - \frac{1}{t_c})^L T_i(k) \\ &+ \sum_{l=0}^{L-1} (1 - \frac{1}{t_c})^{L-l} \hat{R}_i(k+l|k) \delta(i - (\mathbf{i})_{l+1}) \end{aligned} \quad (16)$$

equals $T_i(k+L)$ given that the rate predictions are perfect and that user $(\mathbf{i})_l$ is scheduled in time slot $k+l-1$ for $l = 1, \dots, L$. This is easily verified by solving (4) as a difference equation. Observe next that

$$\begin{aligned} \mathbf{i}^*(k) &= \arg \max_{\mathbf{i} \in \mathcal{F}} \mathcal{O}_{pf}^{(k)}(\mathbf{i}) \\ &= \arg \max_{\mathbf{i} \in \mathcal{F}} \sum_{i=1}^N \log \tilde{T}_i(k|\mathbf{i}) \\ &= \arg \max_{\mathbf{i} \in \mathcal{F}} \sum_{i=1}^N \log \left((1 - \frac{1}{t_c})^{L-1} \tilde{T}_i(k|\mathbf{i}) \right) \end{aligned} \quad (17)$$

This proves that $\mathbf{i}^*(k)$ maximizes $\mathcal{U}(k+L)$ if the rate predictions are perfect.

B. Proof of Proposition 2

Observe that

$$\tilde{T}_j(k+L|\mathbf{i}^n(k) \stackrel{l}{\leftarrow} i) = \hat{T}_j(k+L|\mathbf{i}^n(k) \stackrel{l}{\leftarrow} 0) \quad (18)$$

for $j \neq i$ and

$$\tilde{T}_j(k+L|\mathbf{i}^n(k) \stackrel{l}{\leftarrow} i) = \hat{T}_j(k+L|\mathbf{i}^n(k) \stackrel{l}{\leftarrow} 0) + \frac{1}{t_c} w_l \hat{R}_j(k+l-1|k) \quad (19)$$

for $j = i$. We can therefore write

$$\mathcal{O}_{pf}^{(k)}(\mathbf{i}^n(k) \stackrel{l}{\leftarrow} i) = \sum_{j=1}^N \log \tilde{T}_j(k | \mathbf{i}^n(k) \stackrel{l}{\leftarrow} 0) + \log \left(1 + \frac{\hat{R}_i(k+l-1|k)}{\tilde{T}_i(k+L | \mathbf{i}^n(k) \stackrel{l}{\leftarrow} 0)} \right) \quad (20)$$

This completes the proof since only the last term in the expression above depends on i and $\log(\cdot)$ is a monotone increasing function.

REFERENCES

- [1] R. Knopp and P. Humblet, "Information capacity and power control in single-cell multiuser communications," in *Proc. IEEE Int. Conf. on Commun., (ICC'95)*, Seattle, WA, June 1995, pp. 331–335.
- [2] P. Viswanath, D. Tse, and R. Laroia, "Opportunistic beamforming using dumb antennas," *IEEE Trans. Inform. Theory*, vol. 48, no. 6, pp. 1277–1294, 2002.
- [3] A. Duel-Hallen, S. Hu, and H. Hallen, "Long-range prediction of fading signals," *IEEE Signal Processing Mag.*, vol. 17, no. 3, pp. 62–75, 2000.
- [4] R. Vaughan, P. Teal, and R. Raich, "Short-term mobile channel prediction using discrete scatterer propagation model and subspace signal processing algorithms," in *Proc. VTS 2000*, Boston, MA, Sept. 2000, pp. 751–758.
- [5] T. Ekman, "Prediction of mobile radio channels, modeling and design," Ph.D. dissertation, Uppsala University, Sweden, 2002. [Online]. Available: <http://www.signal.uu.se/Publications/abstracts/a023.html>
- [6] X. Liu, E. Chong, and N. Shroff, "Opportunistic transmission scheduling with resource-sharing constraints in wireless networks," *IEEE J. Select. Areas Commun.*, vol. 19, no. 10, pp. 2053–2064, 2001.
- [7] S. Borst, "User-level performance of channel-aware scheduling algorithms in wireless data networks," *IEEE/ACM Trans. Networking*, vol. 13, no. 3, pp. 636–647, 2005.
- [8] D. Park, H. Seo, H. Kwon, and B. G. Lee, "Wireless packet scheduling based on the cumulative distribution function of user transmission rates," *IEEE Trans. Commun.*, vol. 53, no. 11, pp. 1919–1929, 2005.
- [9] F. Kelly, "Charging and rate control for elastic traffic," *Eur. Trans. Telecommun.*, vol. 8, pp. 33–37, 1997.
- [10] H. Kushner and P. Whiting, "Convergence of proportional-fair sharing algorithms under general conditions," *IEEE Trans. Wireless Commun.*, vol. 3, no. 4, pp. 1250–1259, July 2004.
- [11] H. Kim and Y. Han, "A proportional fair scheduling for multicarrier transmission systems," *IEEE Commun. Lett.*, vol. 9, no. 3, pp. 210–212, Mar. 2005.
- [12] R. Jain, *The art of computer systems performance analysis*. John Wiley and Sons, 1991.

TABLE I

ROBUST PREDICTION-AWARE SCHEDULING

In each time slot k :

- 1) Update the rate predictions, $\hat{R}_i(k+l|k)$.
 - 2) Find $\mathbf{i}^*(k)$ that maximizes $\mathcal{O}^{(k)}(\mathbf{i})$.
 - 3) Schedule the user given by the first component of $\mathbf{i}^*(k)$, i.e $i^*(k) = (\mathbf{i}^*(k))_1$.
-

TABLE II

ITERATIVE ALGORITHM FOR OBTAINING $\mathbf{i}^*(k)$

1) **Initialization**

To initialize the algorithm let

$$\mathbf{i}^0(k) = (i_2^*(k-1), i_3^*(k-1), \dots, i_L^*(k-1), 1).$$

2) **Main iteration**

At each iteration recompute one component of the scheduling vector. For the $(n+1)$ th iteration let

$$\mathbf{i}^{n+1}(k) = \mathbf{i}^n(k) \stackrel{l}{\leftarrow} i_l^{n+1}(k),$$

where $l = L - (n \bmod L)$ and

$$i_l^{n+1}(k) = \arg \max_{i=1, \dots, N} \mathcal{O}^{(k)}(\mathbf{i}^n(k) \stackrel{l}{\leftarrow} i). \quad (21)$$

3) **Termination**

When $\mathbf{i}^n(k) = \mathbf{i}^{n-L}(k)$ we have $\mathbf{i}^m(k) = \mathbf{i}^n(k)$ for all $m \geq n$ and we have converged to a solution.

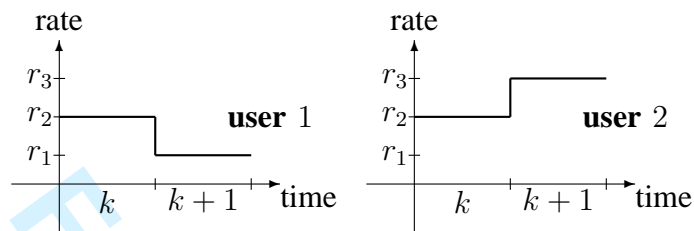


Fig. 1. Supportable rates for user 1 and 2 in time slot k and $k + 1$. To ensure that user 1 is scheduled in time slot k estimates of the supportable rates in time slot $k + 1$ must be available.

$i_1^*(k-1)$	$i_2^*(k-1)$...	$i_{L-1}^*(k-1)$	$i_L^*(k-1)$	$= \mathbf{i}^*(k-1)$	previous solution
$i_2^*(k-1)$	$i_3^*(k-1)$...	$i_L^*(k-1)$	1	$= \mathbf{i}^0(k)$	initialization
$i_2^*(k-1)$	$i_3^*(k-1)$...	$i_L^*(k-1)$	$i_L^1(k)$	$= \mathbf{i}^1(k)$	1st iteration
$i_2^*(k-1)$	$i_3^*(k-1)$...	$i_{L-1}^2(k)$	$i_L^1(k)$	$= \mathbf{i}^2(k)$	2nd iteration
⋮						
$i_1^L(k)$	$i_2^{L-1}(k)$...	$i_{L-1}^2(k)$	$i_L^1(k)$	$= \mathbf{i}^L(k)$	L th iteration
$i_1^L(k)$	$i_2^{L-1}(k)$...	$i_{L-1}^2(k)$	$i_L^{L+1}(k)$	$= \mathbf{i}^{L+1}(k)$	$L + 1$ th iteration

Fig. 2. Illustration of iterative algorithm for obtaining scheduling vector. In each iteration one component is updated with the other components held fixed.

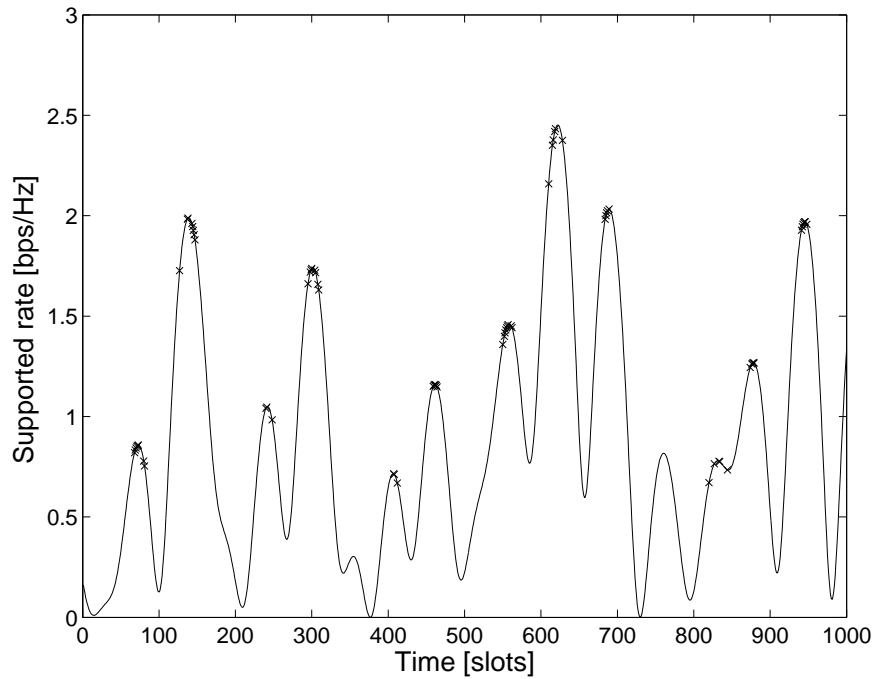
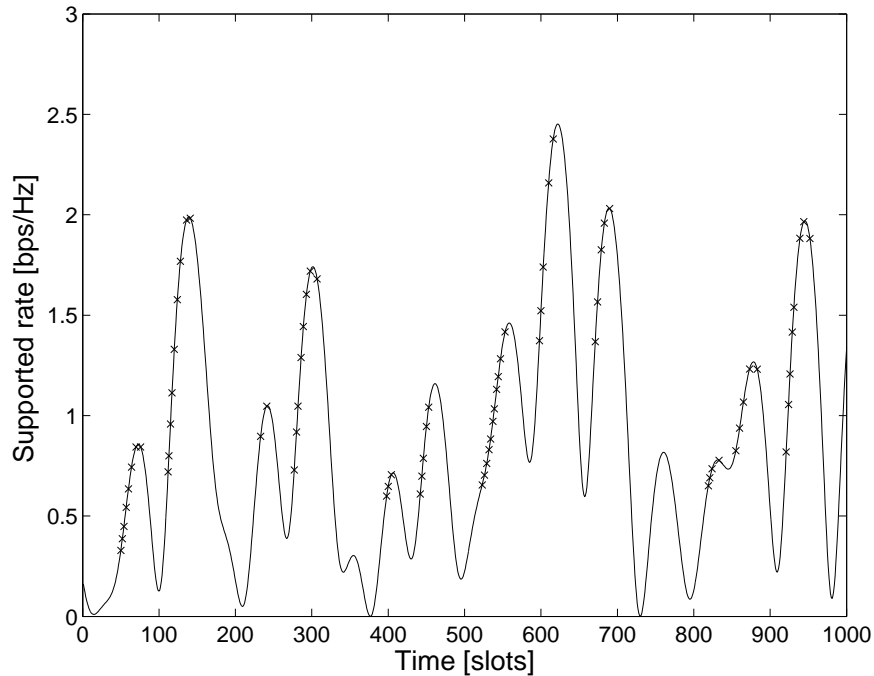
(a) Predictive PFS, $t_c = 100$ and $L = 21$.(b) Standard PFS, $t_c = 100$.

Fig. 3. Snapshot of supported rates for one user. A superimposed cross corresponds to an allocated slot with (a) the predictive (b) the standard PFS algorithm. The predictive algorithm leads to higher throughputs since the allocated slots are more tightly clustered around the fading peaks.

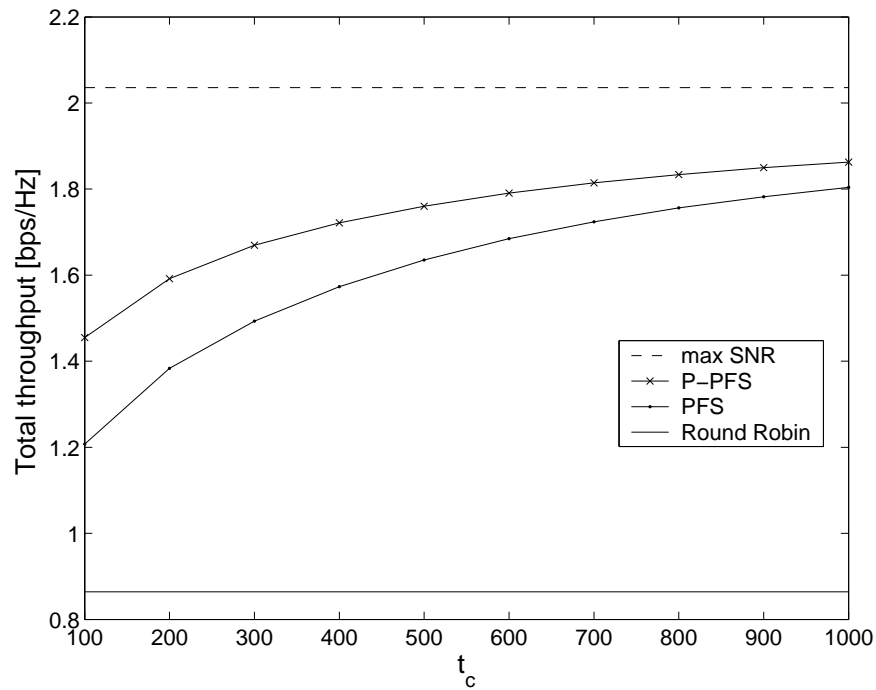


Fig. 4. Total long-term throughput as function of t_c for the predictive and the standard PFS algorithm with 15 users. The prediction horizon $L - 1$ is set to 10 slots for the predictive PFS algorithm.

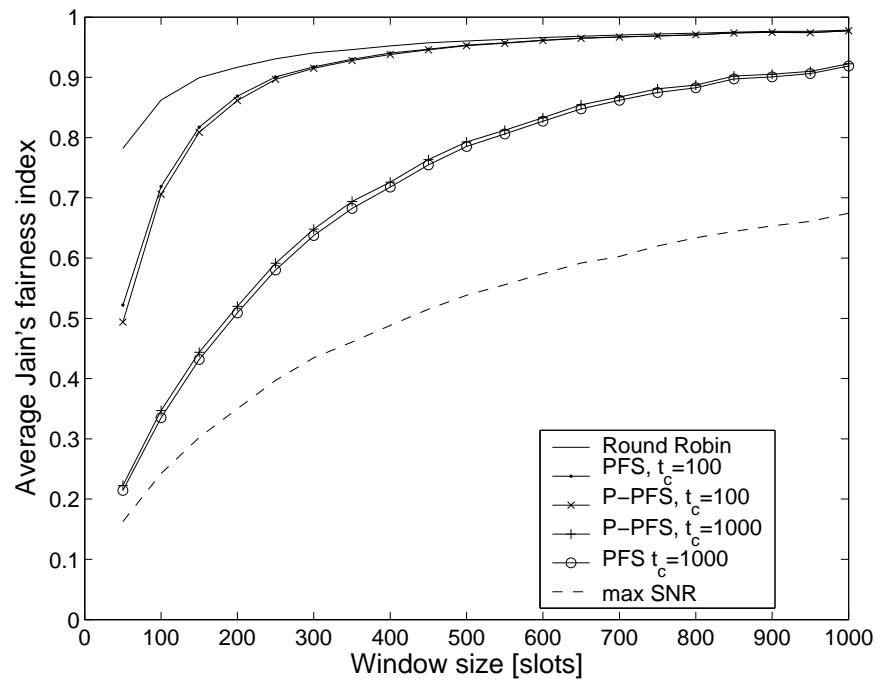
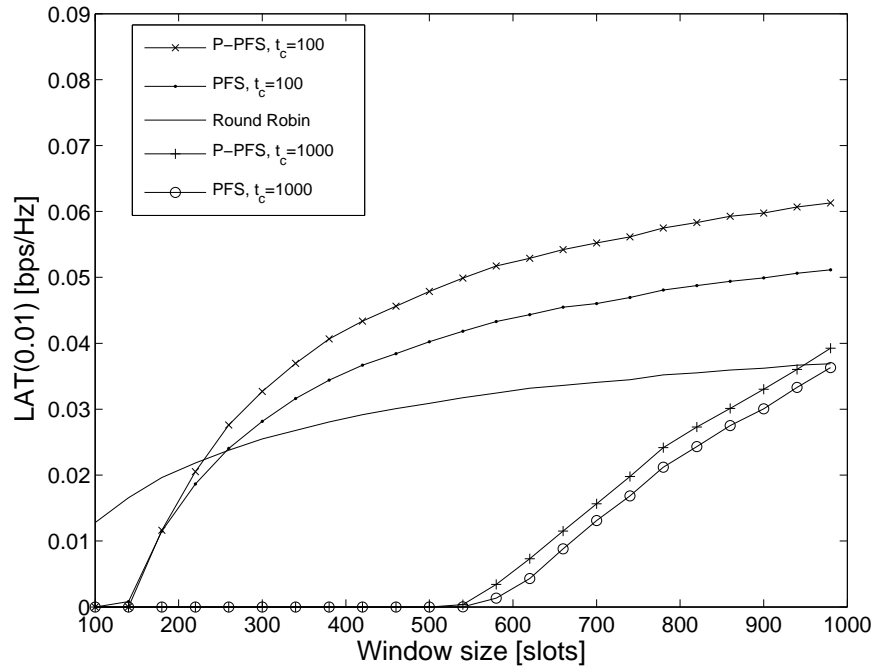
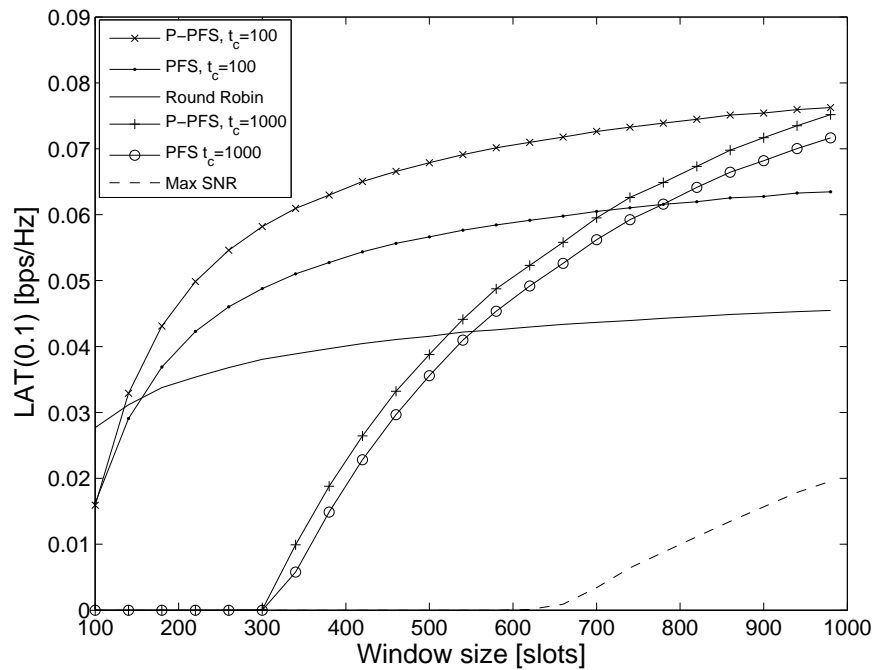


Fig. 5. Jain's fairness index as function of window size for the predictive and the standard PFS algorithm with 15 users and $t_c = 100, 1000$.



(a) LAT(0.01)



(b) LAT(0.1)

Fig. 6. $LAT(\epsilon)$ as a function of window size for (a) $\epsilon = 0.01$ and (b) $\epsilon = 0.1$ in a system with 15 users. The probability that the throughput of a single user falls below $LAT(\epsilon)$ over the specified window size equals ϵ . The prediction horizon $L - 1$ is 10 slots for the predictive PFS algorithm.