Instability of the Proportional Fair Scheduling Algorithm for HDR

Matthew Andrews*

August 28, 2002

Abstract

In this paper we study the Proportional Fair scheduler that has been proposed for scheduling in the High Data Rate (HDR) wireless data system [4, 6]. We consider a single basestation transmitting to a set of mobile users. In each time slot the scheduler has to decide on a mobile to which it will transmit data. The decision is based on information that the basestation receives about the time-varying channels between itself and the mobiles.

We focus on deciding whether or not Proportional Fair is *stable* in a situation with finite queues and a data arrival process. That is, we wish to decide if Proportional Fair keeps all queues bounded whenever this is feasible.

There are in fact multiple versions of Proportional Fair, depending on how it treats small queues. In this paper we consider six different versions and show that all are unstable for one simple example.

Index terms: Wireless scheduling. Proportional Fair. HDR. Instability.

1 Introduction

We consider the Proportional Fair scheduling algorithm for the downlink of a High Data Rate (HDR) wireless system [4, 6]. We assume a single basestation transmitting to a set of mobile users. At each time slot the basestation can transmit to at most one of the mobile users. The purpose of the scheduler is to choose, at each time slot, the mobile to which the basestation should transmit.

*Bell Laboratories, Lucent Technologies, 600 Mountain Avenue, Rm 2C-356, Murray Hill, NJ 07974. `andrews@research.bell-labs.com`

A key feature of this problem is that in a wireless environment, the rate at which the basestation can transmit to the different mobiles is both *mobile-dependent* and *time-dependent*. If a mobile is close to the basestation then the energy required to transmit a bit to the mobile is small and hence the transmission can be at high rates. In contrast, if the mobile is far from the basestation then the energy required to transmit each bit is large and so only low transmission rates are possible. This accounts for the mobile-dependent nature of the transmission rates.

The time-dependent nature of the transmission rates is due to mobility and fading. As a mobile moves away from the basestation the feasible transmission rates decrease. Also, when the channel for a mobile enters a fade, the feasible transmission rates decrease. We shall sometimes refer to the rate at which the basestation can transmit to a mobile as the *channel condition* between the basestation and the mobile.

Motivated by the above properties of wireless networks, we require a scheduling algorithm for a slotted system with user-dependent and time-dependent channel conditions. We assume that the scheduler is aware of the channel conditions for each slot before it has to make the scheduling decision for that slot. In a wireless system this is feasible as long as the mobile users are continually measuring their channel condition and reporting this information back to the basestation. A desirable feature of any scheduling algorithm is that it tends to serve a user when its channel is "good". However, we cannot simply always serve the user with the best channel since then some other users may be starved.

In this paper we study the *Proportional Fair* scheduler that has been proposed for scheduling in the HDR system [4, 6]. In [11] it is shown that under certain conditions Proportional Fair maximizes the sum of the logarithms of the long-run average data rates provided to the users. The reason that this is a desirable objective is that we then achieve "fairness" in the sense that increasing some user's rate by a multiplicative factor has the same effect on the objective as multiplying another user's rate by the same multiplicative factor.

Most of the analysis of wireless schedulers, and in particular the analysis of Proportional Fair, assumes *infinitely backlogged queues*. That is, we assume that at all times all mobiles have data queued at the basestation that needs to be transmitted to them.

However, in reality data will arrive at the basestation for each mobile according to a data arrival process. Hence at each time slot there may be some mobiles for which there is no data. Our main goal now is to make sure that all data is transmitted in a bounded amount of time. That is, we want to keep the queues *stable*. In this paper we show that the Proportional Fair algorithm is not stable in some simple situations. One feature of the Proportional Fair algorithm is that if queues are not infinitely backlogged then there are multiple possible definitions of the algorithm depending on how it deals with queues with little or no data. In this paper we define six different versions of Proportional Fair and show that all six can be unstable.

The model

Our model is motivated by the HDR system for high speed data transmission in wireless networks. We assume a basestation transmitting data to multiple mobile users. Time is divided into *time slots*. In each time slot the basestation can transmit to at most one user. For each time slot, each mobile user sends a *Data Rate Control* (DRC) message to the basestation that indicates the rate at which the basestation can transmit to that user if it is selected. We denote the amount of data that can be transmitted to user i in time slot t by $DRC_i(t)$.

Each user has a queue that is fed by a data arrival process. We use $a_i(t)$ to denote the amount of data that arrives for user i in time slot t. We use $q_i(t)$ to denote the queue size for user i in time slot t. We use $x_i(t)$ to denote the scheduling decision made at time t. If user i is chosen at time t then we set $x_i(t)$ equal to 1. We set $x_i(t)$ equal to 0 otherwise. If the basestation decides to schedule data to mobile i in time slot then the amount of data that it can send is $\max\{DRC_i(t), q_i(t)\}$, i.e. the basestation cannot send more data than the DRC value and it cannot send more data than is in the queue. We note that the data arrival process $\{a_i(t)\}$ and the DRC process $\{DRC_i(t)\}$ completely define the input to our system.

Our aim is to provide system *stability*, i.e. we wish to keep the queues bounded over time. However, in order to achieve this we must put some restrictions on the input to the system. We say that the system is *admissible* if the data arrival process and the DRC process have the property that there *exists* some schedule under which all the queues are bounded. We say that a scheduler is *stable* if it keeps the queues bounded whenever the system is admissible. In this paper we show that the Proportional Fair scheduler is *unstable*. We now define Proportional Fair.

The basic definition of Proportional Fair is that in each time slot it picks the user that maximizes

$$\frac{DRC_i(t)}{R_i(t)},$$

where $R_i(t)$ is an *exponentially smoothed average* of the service rate received by user i. The exact definition of $R_i(t)$ depends on the version of Proportional Fair that we are using. We elaborate below. The Proportional Fair algorithm has desirable properties in that in general it favors users that have a high DRC value. This helps to keep the system throughput high. However, if user i is not receiving any service, the value of $1/R_i(t)$ increases and so we are more likely to serve user i in the next slot.

For the case of infinitely backlogged queues (i.e. $q_i(t) = \infty$) this completes the definition of Proportional Fair. However, for the case of finite queues there are different versions depending on which users are eligible for service and on how we update the exponentially smoothed service rate $R_i(t)$.

More precisely, there are three possible methods for deciding who is eligible to compete.

1. All users get to compete for a slot.
2. User i only gets to compete for a slot if it has a non-empty queue, i.e. $q_i(t) > 0$.
3. User i only gets to compete for a slot if it can use the entire slot, i.e. $q_i(t) \geq DRC_i(t)$.

There are 2 possible methods for updating the exponentially smoothed average service rate, $R_i(t)$. We can either update based on the value of the DRC or we can update based on how many bits are actually served. That is, we can update the average for user i based on how much service user i is offered or we can update based on how much of this service is used by the user. More precisely, the first method is given by,

$$R_i(t+1) = (1-\tau)R_i(t) + \tau x_i(t) DRC_i(t),$$

The second method is given by,

$$R_i(t+1) = (1-\tau)R_i(t) + \tau x_i(t) \min(q_i(t), DRC_i(t)).$$

The parameter τ is the inverse of the time constant for the exponential filter. Throughout this writeup we take $\tau = 1/1000$ (the value suggested in [6]).

Combining these three methods for deciding who is eligible with the two methods for updating average rate we get six possible versions of Proportional Fair[1]. However, in our instability examples data will arrive to all users in every slot. Therefore $q_i(t)$ will never be zero and hence methods 1 and 2 for deciding who is eligible are the same. Moreover, if we use method 3 for deciding who is eligible then the two methods for updating the average rate are the same. Therefore we only need to consider three versions of Proportional Fair. We remark that we shall use the same instability example for all three versions.

In order to make our analysis consistent we must define the exact sequence of events in a time slot. These are,

(a) Data arrives for each user.

(b) $q_i(t)$ is updated based on the new data for user i and the amount of service user i received in slot $t-1$.

(c) $R_i(t)$ is updated based on the amount of service user i received in slot $t-1$.

(d) A scheduling decision is made for slot t and data is served.

There are two possible definitions of instability with respect to how the initial configuration is defined. We can either say that only we need to show there *exists* an initial configuration that leads to instability or we can say that we need to show instability starting from an empty system. In our proofs we satisfy both definitions. We first define a convenient initial condition from which we can show instability. We then show that we get instability from the *empty initial configuration* in which $q_i(0) = 0$ for all i and $R_i(0) = \tau$ for all i. (Note that we cannot take $R_i(0) = 0$ initially since then we would not able to evaluate $DRC_i(0)/R_i(0)$ when scheduling the first slot.) For the latter result we simply show using a computer calculation that after a finite amount of time, the system reaches the convenient initial configuration used by the former result.

[1]One could think of additional versions in which we have different rules for the case in which *no* user is eligible. However, in our instability examples there will always be at least one eligible user. Hence all such additional versions are covered by our analysis.

Previous work

The Proportional Fair scheduler is presented by Jalali et al.[6] in the context of the HDR wireless data system. Tse [11] showed that under certain conditions the Proportional Fair scheduler maximizes the sum of the logarithms of the rates experienced by the users.

Although we show in this paper that Proportional Fair is unstable, it is possible to achieve stability in our setting. If the data arrival process and the DRC process are ergodic stochastic processes then algorithms known as MAX-WEIGHT [10, 1, 2], MAX-DELAY [1, 2], and EXP [8, 9] are known to be stable. The MAX-WEIGHT algorithm always serves the user that maximizes $q_i(t)DRC_i(t)$ and the MAX-DELAY algorithm always serves the user that maximizes $q_i(t)\Delta_i(t)$, where $\Delta_i(t)$ is the Head-of-Line delay for user i. The EXP algorithm is a more complex algorithm that in addition to stability tries to keep user delays in desired proportions. For the case of adversarial systems in which the data arrival process and DRC process have no ergodic properties, a "Tracking Algorithm" is defined in [3] that is able to achieve stability whenever possible.

In [7], Shakkottai and Srikant consider a situation in which all DRC values are either 0 or 1 and each packet has a deadline. They discuss the problem of meeting these deadlines. In [5], Borst and Whiting focus on the problem of meeting a user-defined set of target rates.

2 Proof of Instability of Proportional Fair

Instability Example

As mentioned in Section 1, although we defined six versions of Proportional Fair, as long as we make sure that all users have data at all times then it is sufficient to consider only three of the versions. We now define the instability example which is the same for all three versions. We consider 2 users, A and B. The arrival rates are constant and the channel rates are periodic with period 10.

- In each slot 94 bits arrive for user A.
- In each slot 49 bits arrive for user B.
- In all slots, the DRC for user A is 100 bits per slot.

- In slot $10n$ (for integer n), the DRC for user B is 1000 bits per slot. In slot $10n+i$, $1 \le i \le 9$, the DRC for user B is 100 bits per slot.

We say that a slot is "B-biased" if the DRC for user B in that slot is higher than the DRC for user A. It can be seen that 10% of the slots are B-biased. Note that traffic arrives for both users in every slot and so both queues are non-zero whenever a scheduling decision is made.

We must first show that there exists an algorithm that keeps these queues stable. We then show that under three versions of Proportional Fair, the queue lengths are unbounded. Suppose that half of the B-biased slots are assigned to user B and all other slots are assigned to user A. This means that 95% of all slots are assigned to user A and so the service rate for user A is 95 bits per slot. The remaining slots are all B biased and so the service rate for user B is 50 bits per slot. Hence there exists a schedule that keeps the queues stable.

Instability of Proportional Fair: Version 1

For this case,

- Average rate is updated according to $R_i(t+1) = (1-\tau)R_i(t) + \tau x_i(t) DRC_i(t)$.

- All users get to compete.

We now show that this version of Proportional Fair is unstable. Suppose that there exists a B-biased slot T_0 such that after this slot the average rates satisfy $89 < R_A(T_0) < 90$ and $100 < R_B(T_0) < 101$. We can either enforce this at time $T_0 = 0$ through the initial conditions or else we can observe via computer simulation that this occurs at time $T_0 = 5400$ when we start from the empty initial configuration (defined in Section 1). We prove instability by induction.

Suppose inductively that after the B-biased slot t we have, $89 < R_A(t+1) < 90$ and $100 < R_B(t+1) < 101$. By definition this is true for $t = T_0$. We first show that slots $t+1, \ldots, t+9$ are assigned to user A and the B-biased slot $t+10$ is assigned to user B. Clearly, $R_A(t+1) < R_B(t+1)$. Recall that $\tau = 1/1000$. For $2 \le i \le 9$ we have,

$$
\begin{aligned}
R_A(t+i) &< 90(1-\tau)^{i-1} + \sum_{j=2}^{i} 100\tau(1-\tau)^{j-2} < 91, \\
R_B(t+i) &> 100(1-\tau)^{i-1} > 99.
\end{aligned}
$$

In addition we have $DRC_A(t+i) = DRC_B(t+i)$ for these slots. Hence slots $t+1,\ldots,t+9$ are all assigned to user A. For slot $t+10$ we have,

$$\begin{aligned} R_A(t+10) &> 89(1-\tau)^9 > 88, \\ R_B(t+10) &< 101(1-\tau)^9 + \sum_{j=2}^{10} 100\tau(1-\tau)^{j-2} < 101. \end{aligned}$$

In addition we have $DRC_B(t+10) = 10 * DRC_A(t+10)$. Therefore, slot $t+10$ is assigned to user B.

We have now characterized how the algorithm assigns slots $t+1,\ldots,t+10$. It remains to bound $R_A(t+11)$ and $R_B(t+11)$ so that we can use induction to determine how all time slots are assigned. We have,

$$\begin{aligned} R_A(t+11) &= R_A(t+1)(1-\tau)^{10} + \sum_{j=1}^{9} 100\tau(1-\tau)^j \\ &= R_A(t+1)(1-\tau)^{10} + 100(1-\tau-(1-\tau)^{10}), \\ R_B(t+11) &= R_B(t+1)(1-\tau)^{10} + 1000\tau. \end{aligned}$$

It can be verified that since $\tau = 1/1000$ then the mapping from $R_A(t+1)$ to $R_A(t+11)$ maps the interval $(89, 90)$ into a subset of itself and the mapping from $R_B(t+1)$ to $R_B(t+11)$ maps the interval $(100, 101)$ into a subset of itself. Therefore $89 < R_A(t+11) < 90$ and $100 < R_B(t+11) < 101$. By induction user A is only assigned 90% of the slots. Therefore the queue for user A is unstable.

Instability of Proportional Fair: Version 2

For this case,

- Average rate is updated according to $R_i(t+1) = (1-\tau)R_i(t) + \tau x_i(t)\min(q_i(t), DRC_i(t))$.
- All users get to compete.

We now prove that this version of Proportional Fair is unstable. Suppose that there exists a time slot T_0 that is B-biased and satisfies $10R_A(T_0) > R_B(T_0)$, $q_B(T_0) < 1000$ and $q_A(T_0) > 2000$. We can either enforce this at time $T_0 = 0$ through the initial conditions or else we can observe via computer simulation that this occurs at time $T_0 = 50$ when we start from the empty initial configuration. We prove instability via contradiction.

The time slot T_0 will certainly be assigned to user B since $10R_A(T_0) > R_B(T_0)$ and $DRC_B(T_0) = 10DRC_A(T_0)$. This implies $q_B(T_0+1) = 49$. We now show that all future B-biased time slots are assigned to user B.

Suppose not. Let $t \geq T_0 + 10$ be the first B-biased time slot after slot T_0 that is not assigned to user B. This means that $10R_A(t) \leq R_B(t)$. However, since t is the first such slot we also have $10R_A(t-10) \geq R_B(t-10)$, $q_B(t-9) = 49$ and $q_A(t-9) > 2000$. (Since every B-biased slot is assigned to user B these slots are able to empty the queue for user B. The number 49 comes from the user B arrivals in slot $t-9$. In addition, queue A has been increasing since time T_0.) There are 2 cases to consider.

Case 1. Slots $t-9, \ldots, t-1$ are assigned to user B. This implies that $q_B(t-9) = \ldots = q_B(t-1) = 49$. This in turn implies that $R_B(t-1) = (1-\tau)^9 R_B(t-10) + 49\sum_{j=0}^{8}\tau(1-\tau)^j$ and $R_B(t) = (1-\tau)^{10}R_B(t-10) + 49\sum_{j=0}^{9}\tau(1-\tau)^j$. But we also have $10R_A(t) \leq R_B(t)$, $R_A(t) = (1-\tau)R_A(t-1)$ and $R_A(t-1) \geq R_B(t-1)$. Hence we have,

$$\begin{aligned}
R_A(t-1) &\geq (1-\tau)^9 R_B(t-10) + 49(1-(1-\tau)^9),\\
\Rightarrow R_A(t) &\geq (1-\tau)^{10} R_B(t-10) + 49(1-\tau-(1-\tau)^{10}),\\
\text{and } R_A(t) &\leq \frac{1}{10}(1-\tau)^{10} R_B(t-10) + \frac{49}{10}(1-(1-\tau)^{10}).
\end{aligned}$$

Hence,

$$\frac{1}{10}(1-(1-\tau)^{10}) \geq (1-\tau-(1-\tau)^{10}),$$

which is not true since $\tau = 1/1000$. Therefore we have a contradiction.

Case 2. Slot $t-i$ is assigned to user A for some $1 \leq i \leq 9$. Since $q_A(t-9) > 2000$ we know that user A will be able to fully utilize the slot. We have,

$$\begin{aligned}
R_A(t) &\geq (1-\tau)^{10} R_A(t-10) + 100\tau(1-\tau)^{i-1},\\
\Rightarrow 10R_A(t) &\geq (1-\tau)^{10} R_B(t-10) + 1000\tau(1-\tau)^8.
\end{aligned}$$

In addition,

$$R_B(t) \leq (1-\tau)^{10} R_B(t-10) + 490\tau,$$

since the total number of bits available to be serviced by user B during time slots $t-10, \ldots, t-1$ is at most 490. Hence $10R_A(t) > R_B(t)$ for $\tau = 1/1000$. This contradicts our assumption that $10R_A(t) \leq R_B(t)$.

In both cases we have a contradiction. This implies that after time T_0 all B-biased slots are assigned to user B. This means that the queue for user A is serviced at rate at most 90 bits per slot which implies that the queue for user A is unstable.

Instability of Proportional Fair: Version 3

For this case,

- Average rate is updated according to $R_i(t+1) = (1-\tau)R_i(t) + \tau x_i(t) DRC_i(t)$.
- User i only gets to compete for a slot if it can use the entire slot, i.e. $q_i(t) \geq DRC_i(t)$.

We now show that this current version of Proportional Fair is unstable. Suppose that there exists a time T_0 for which the following conditions hold,

- $R_A(T_0) > 50$, $R_B(T_0) < 49$.
- $q_A(T_0) > 2000$, $q_B(T_0) < 100$.

We can either enforce this at time $T_0 = 0$ through the initial conditions or else we can observe via computer simulation that this occurs at time $T_0 = 3951$ when we start from the empty initial configuration.

Suppose inductively that during the interval $[T_0, T_0 + k]$, user B is assigned all the slots that are *not* B-biased and that satisfy $q_B(t) \geq 100$. This is vacuously true for $k = 0$. We now prove that this holds for the interval $[T_0, T_0 + k + 1]$.

Lemma 1 *During the interval* $[T_0, T_0 + k]$*, the maximum latency of any user* B *bit is* 5 *time slots.*

Proof: If q_B becomes larger than 100 in a slot that is not B-biased then by our inductive assumption q_B is immediately reduced to below 100. (Recall that 49 bits arrive for user B in every time slot.) If q_B is becomes larger than 100 in a slot that is B-biased then user B might not be served. However, user B will be served in the next slot (by the periodic nature of the channel conditions).

The above implies that $q_B \leq 198$ during the interval $[T_0, T_0 + k]$. However, it can easily be seen that if any bit has latency more than 5 time slots then q_B would become larger than 198. □

Lemma 2 $R_B(T_0 + k + 1) < 49.3$.

Proof: Let X be the set of user B bits that were in the queue at time T_0 or that arrived during the interval $[T_0, T_0 + k]$. For any bit b in X let a_b be the time slot in which it arrived and let d_b be the time slot in which it was served. The above argument implies that $a_b \leq d_b \leq a_b + 5$. By examining how the service of each bit contributes to the average rate we have,

$$\begin{aligned} R_B(T_0 + k + 1) &= (1-\tau)^{k+1} R_B(T_0) + \sum_{b \in X, d_b \in [T_0, T_0+k]} \tau (1-\tau)^{T_0+k-d_b} \\ &\leq (1-\tau)^{k+1} R_B(T_0) + \sum_{b \in X} \tau (1-\tau)^{T_0+k-a_b-5} \\ &= (1-\tau)^{k+1} R_B(T_0) + \sum_{t \in [T_0, T_0+k]} 49\tau (1-\tau)^{T_0+k-t} (1-\tau)^{-5} \\ &< 49(1-\tau)^{k+1} + 49(1-(1-\tau)^{k+1})(1-\tau)^{-5} \\ &< 49(1-\tau)^{-5} \\ &< 49.3, \end{aligned}$$

since $\tau = 1/1000$. □

Lemma 3 $R_A(T_0 + k + 1) > 49.7$.

Proof: By the inductive assumption, during the interval $[T_0, T_0 + k]$, user B uses every slot that is not B-biased and for which q_B is at least 100. It is easy to see therefore that user B is assigned at least 1 slot in every 10. Hence the service rate for user A is always at most 90 bits per slot. By comparing with the arrival rate of user A and the fact that $q_A(T_0)$ is at least 2000 we can see that the queue for user A is always at least 1100 during the interval $[T_0, T_0 + k]$. Therefore user A can utilize any slot that is not assigned to user B.

Therefore in any slot the number of bits assigned to user A is exactly 100 minus the number of bits assigned to user B. We have,

$$\begin{aligned} R_A(T_0 + k + 1) &= (1-\tau)^{k+1} R_A(T_0) + \sum_{t \in [T_0, T_0+k]} 100\tau (1-\tau)^{T_0+k-t} \\ &\quad - \sum_{b \in X, d_b \in [T_0, T_0+k]} \tau (1-\tau)^{T_0+k-d_b} \\ &\geq (1-\tau)^{k+1} R_A(T_0) + \sum_{t \in [T_0, T_0+k]} 100\tau (1-\tau)^{T_0+k-t} - \sum_{b \in X} \tau (1-\tau)^{T_0+k-a_b-5} \\ &= (1-\tau)^{k+1} R_A(T_0) + \sum_{t \in [T_0, T_0+k]} \tau (1-\tau)^{T_0+k-t} (100 - 49(1-\tau)^{-5}) \end{aligned}$$

$$
\begin{aligned}
&> 50(1-\tau)^{k+1} + (1-(1-\tau)^{k+1})(51 - 49((1-\tau)^{-5} - 1)) \\
&> 50 - 49((1-\tau)^{-5} - 1) \\
&> 49.7,
\end{aligned}
$$

since $\tau = 1/1000$. □

We are now ready to prove the inductive hypothesis for time slot $T_0 + k + 1$. If time slot $T_0 + k + 1$ is B-biased or $q_B(T_0 + k + 1) < 100$ then there is nothing to prove. If the slot is not B-biased and $q_B(T_0 + k + 1) \geq 100$ then the slot is assigned to user B because $DRC_A(T_0 + k + 1) = DRC_B(T_0 + k + 1)$, $R_B(T_0 + k + 1) < 49.3 < 49.7 < R_A(T_0 + k + 1)$ and user B is eligible for the slot.

Therefore by induction user B is assigned all slots after time T_0 that are not B-biased and for which $q_B \geq 100$. This trivially implies that user B receives at least one slot in every 10 slots. Hence user A is served at rate at most 90 bits per slot which is not enough to keep its queue stable.

3 Conclusions

In this paper we showed that six natural versions of the Proportional Fair algorithm can be unstable. We believe that this result is of interest since Proportional Fair has been proposed as the scheduling algorithm for the HDR wireless data system. Some open questions remain. Although stable scheduling algorithms are known (e.g. [1, 2, 3]), more works needs to be done to provide optimal or near-optimal delay bounds. In particular, if all data has a *deadline* by which it needs to served, it would be interesting to know if there is a scheduler that can maximize the number of satisfied deadlines.

Acknowledgements

The author wishes to thank Sasha Stolyar and Sem Borst for many helpful discussions regarding the performance of Proportional Fair.

References

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

[2] M. Andrews, K. Kumaran, K. Ramanan, A. Stolyar, R. Vijayakumar, and P. Whiting. Providing quality of service over a shared wireless link. *IEEE Communications Magazine*, February 2001.

[3] M. Andrews and L. Zhang. Scheduling over a time-varying user-dependent channel with applications to high speed wireless data. In *Proceedings of the 43rd Annual Symposium on Foundations of Computer Science*, Vancouver, Canada, November 2002.

[4] P. Bender, P. Black, M. Grob, R. Padovani, and N. Sindhushayana A. Viterbi. CDMA/HDR: A bandwidth efficient high speed data service for nomadic users. *IEEE Communications Magazine*, July 2000.

[5] S. Borst and P. Whiting. Dynamic rate control algorithms for CDMA throughput optimization. In *Proceedings of IEEE INFOCOM '01*, Anchorage, AK, April 2001.

[6] A. Jalali, R. Padovani, and R. Pankaj. Data througput of CDMA-HDR: A high efficiency-high data rate personal communication wireless system. In *Proceedings of the IEEE Semiannual Vehicular Technology Conference, VTC2000-Spring*, Tokyo, Japan, May 2000.

[7] S. Shakkottai and R. Srikant. Scheduling real-time traffic with deadlines over a wireless channel. In *Proceedings of ACM Workshop on Wireless and Mobile Multimedia*, Seattle, WA, August 1999.

[8] S. Shakkottai and A. Stolyar. Scheduling algorithms for a mixture of real-time and non-real-time data in HDR. In *Proceedings of 17th International Teletraffic Congress (ITC-17)*, pages 793 – 804, Salvador da Bahia, Brazil, 2001.

[9] S. Shakkottai and A. Stolyar. Scheduling for multiple flows sharing a time-varying channel: The exponential rule. *Analytic Methods in Applied Probability*, 2002. To appear.

[10] L. Tassiulas and A. Ephremides. Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks. *IEEE Transactions on Automatic Control*, 37(12):1936 – 1948, December 1992.

[11] D. Tse. Forward-link multiuser diversity through rate adaptation and scheduling. Submitted to *IEEE Journal on Selected Areas in Communications*.